Vibration Suppression in Finite Length Marine Cable Systems

by

Christopher R. Levesque

//

B.S., Mechanical Engineering, Rensselaer Polytechnic Institute, 1987

Submitted to the Department of Ocean Engineering
and the Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degrees of

Naval Engineer
and
Master of Science in Mechanical Engineering

at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
September 1997

Vibration Suppression in Finite Length Marine Cable Systems

by

Christopher R. Levesque

Submitted to the Department of Ocean Engineering
and
the Department of Mechanical Engineering
on September 15, 1997 in partial fulfillment of the
requirements for the degrees of
Naval Engineer
and
Master of Science in Mechanical Engineering

Abstract

The vibration suppression effectiveness of a flexible in-line marine cable vibration absorber is studied. The transfer matrix method is used to build various numerical models of vibration absorbers in marine cable systems. The models determine cable system natural frequencies, mode shapes and modal damping ratios.

The introduction of absorber damping is shown to result in complex roots to the modal characteristic equations. A computer complex root solver is used to solve for the complex roots of the characteristic equations, resulting in complex system natural frequencies. The significance of complex natural frequencies is explained. Complex natural frequencies are used to calculate modal damping ratios.

The models demonstrate that absorber effectiveness is heavily dependent on absorber location, absorber mass and absorber length. Parametric variation is used to achieve maximum effectiveness of the flexible in-line absorber. Even under optimum conditions, it is shown that the absorber provides insufficient damping to reduce vortex-induced vibrations in water.

The same transfer matrix method is used to evaluate the effectiveness of a mass-spring-dashpot type absorber in a marine cable system. This type of absorber is shown to produce adequate damping to reduce vortex-induced vibrations in water. The transfer matrix method used in this thesis is validated by analyzing the same system using an approach by Den Hartog [1]. The transfer matrix approach combined with complex root solving capability is shown to provide an effective analysis method for marine cable systems.

Thesis Supervior: J. Kim Vandiver
Title: Professor of Ocean Engineering

Thesis Reader: Ain A. Sonin
Title: Professor of Mechanical Engineering

Acknowledgments

I would like to thank Professor J. Kim Vandiver, my thesis supervisor, for his guidance and teaching. His support throughout my thesis effort has made it an enlightening and enjoyable experience. I would also like to thank Doctor Rama Rao for his insight, helping me through several stalling points during my work.

I would also like to recognize my wife, Jackie, and children, Michael, Aimee and Jared. They are also my teachers, from whom I continue to learn more about myself and more about the world with each passing day.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Long slender marine structures such as cables and pipelines are subject to a condition called vortex induced vibration. This occurs when the cylindrical structure is subjected to a cross flow current as shown in figure 1-1:

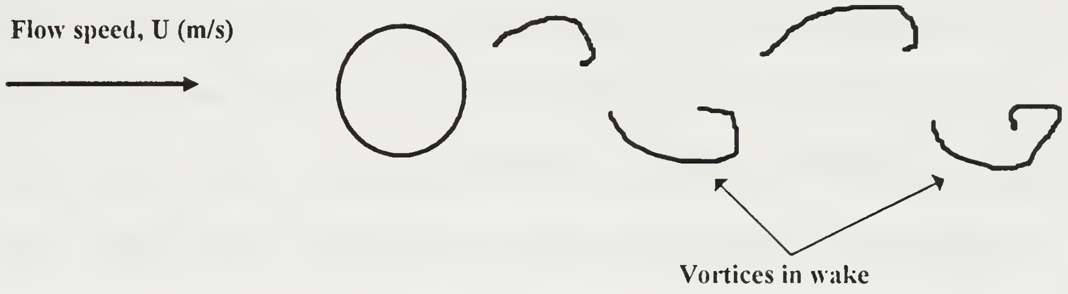


Figure 1-1: Vortex Shedding

At a critical value of flow speed, U, a periodic shedding of vortices occurs on the upper and lower surfaces of the cylinder. This behavior is known as a Von Karman street. As each vortex is shed into the wake, a localized pressure drop occurs. The result is a harmonically varying vertical (lift) force on the cylinder. The frequency at which vortex shedding occurs is related to cylinder diameter and flow velocity by a dimensionless parameter called the Strouhal number:

$$St = \frac{f\ D}{U} \qquad (1 - 1)$$

where: f = vortex shedding frequency (Hz)

D = cylinder diameter (m)

U = flow speed (m/sec)

The Strouhal number is approximately 0.2 for a wide range of Reynolds numbers. Vortex-induced vibrations are especially severe when the vortex shedding frequency is near a resonance frequency of the structure. Cross-flow displacement of the cylinder at this "lock-in" frequency may be on the order of one cylinder diameter. Such vibrations can fatigue marine structures and cause eventual failure.

This problem motivates the investigation into a dynamic absorber for long cylindrical marine structures. Before analyzing the effect of various damping devices, it is worth while to determine the magnitude of damping which must be introduced into a marine structure to cause significant reduction in vortex induced vibrations. In figure 1-2 below, Griffin [13] shows how cross flow displacement varies with a dimensionless parameter called the reduced damping, S_G. Reduced damping is the ratio of damping force to fluid exciting force.

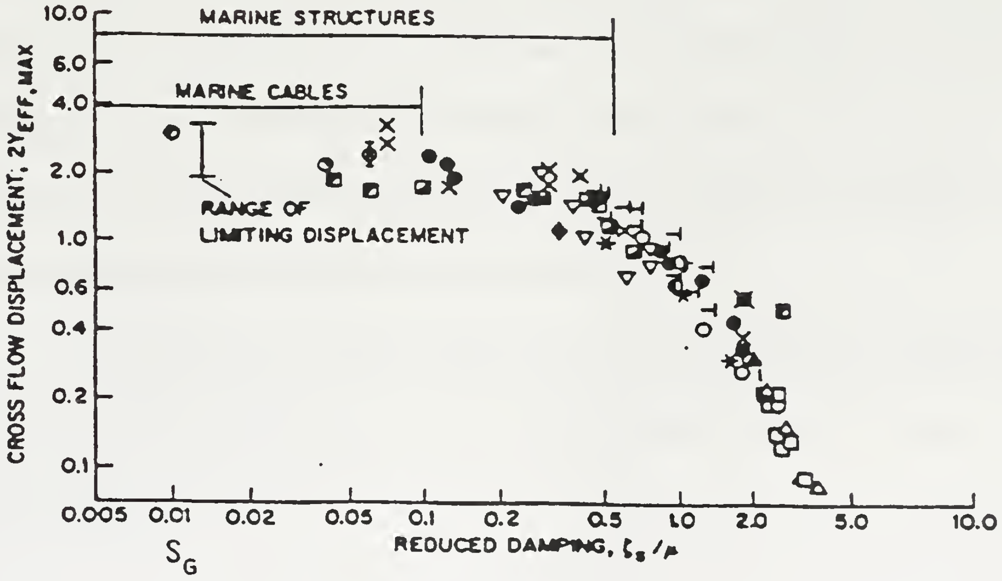


Figure 1-2: Vibration Displacement vs. Reduced Damping

where: $S_G = 8\,\pi^2\, S_t^2\, \zeta\, (\, m_s \,/\, \rho_f\, D^2) \equiv$ reduced damping (1-2)

S_t = Strouhal number

m_s = linear density of cable (kg/m)

ρ_f = fluid density (kg/m^3)

D = cable diameter (m)

The reason displacement amplitude approaches a plateau value at low reduced damping (values below S_G about 0.2) is due to the self-limiting nature of vortex induced vibrations. At values of displacement amplitude greater than approximately one cylinder diameter, the extreme motion of the cylinder begins to disturb the formation of the Von Karman street. Stuctural damping is not the limiting factor at low values of reduced damping. The reduced damping is valuable in determining the amount of damping which must be added to the structure to have an effect on displacement amplitude. It would not be worthwhile to add damping such that reduced damping would not exceed the "knee in the curve" value of 0.2 illustrated in figure 1-2. This is also a very convenient approach since the damping ratio used to calculate reduced damping is the damping which one would measure in an in vacuo transient decay test. Thus, various absorber configurations can be evaluated without the need to consider hydrodynamic effects.

This thesis will study various vibration absorber configurations for marine cables. Most of the analysis will focus on the type of absorber shown in figure 1-3 on the following page:

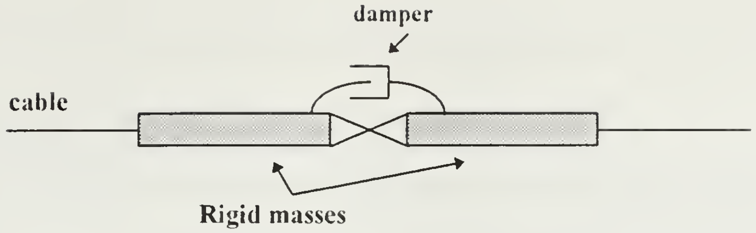


Figure 1-3: Cable Vibration Absorber

This absorber was first proposed by Vandiver and Li [10] . Energy dissipation occurs by way of relative angular motion at the pivot between the two rigid masses. The damping device is integral to the pivot but it is shown schematically in figure 1-3. Butler [7] showed this type of damping device to be possible for marine cable applications. It was seen as a very versatile configuration for marine cables since such an absorber could be clamped on at a desired location while deploying a cable. Or, if the absorber were permanently installed, its size and orientation might permit the absorber to be reeled onto a cable drum when the cable is not deployed.

Chapter 2 describes the transfer matrix approach for analyzing multi-component mechanical systems *with finite boundaries*. It is a very convenient method for studying the dynamic behavior of marine cable systems with attached damping devices. The analysis utilizes the absorber transfer matrix derived by Li [5].

In chapter 3, the transfer matrix method is used to determine natural frequencies, mode shapes and damping ratios of the cable / absorber system. A method is formulated for determining the optimum value of damping constant.

In chapter 4 , system parameters such as absorber length and mass are varied to study their effect on damping performance. Also, use of multiple absorbers and the effect of absorber location relative to vibration nodes and antinodes was studied. Use of a more

conventional absorber type is studied by merging the transfer matrix approach with a more classical approach (Den Hartog [1]) with some interesting results.

Chapter 5 concludes the study and suggests areas for further research.

Chapter 2

The Transfer Matrix Method

The transfer matrix method allows for a convenient matrix representation of the marine cable system. In 1963, Pestel and Leckie [2] recognized that the introduction of numerical computing made matrix methods a powerful tool for studying mechanical system dynamics. In marine cable systems, features such as rigid bodies and discrete damping devices are well suited to matrix modeling techniques. Today's computing speed allows complicated marine cable systems to be modeled with relative ease.

2.1 The Marine Cable System

2.1.1 Assumptions

It is important here to state the assumptions employed in representing the cable system in this way. These assumptions are those of the classic linear string wave equation. They were used by Li [5] in developing the transfer matrices used in this analysis:

1) the cable systems are taut and undergo only small deflections

2) cable bending stiffness is neglected

3) the masses in the cable system are rigid

4) the weight of these masses is small compared to cable tension

5) the dynamic variation in tension in the cable is small and neglected

6) only planar motion is considered (2 dimensional)

2.1.2 Transfer Matrix Representation of a Marine Cable System

Transfer matrices are used to relate the physical state of one location in a mechanical system to that at another location in the system. The state at any location in the system is represented by the state vector. In a cable system, the state at any location can be represented by the state vector, z:

$$[z]=\begin{bmatrix}\overline{y}(x)\\ T\overline{y}'(x)\end{bmatrix} \qquad (2\text{-}1)$$

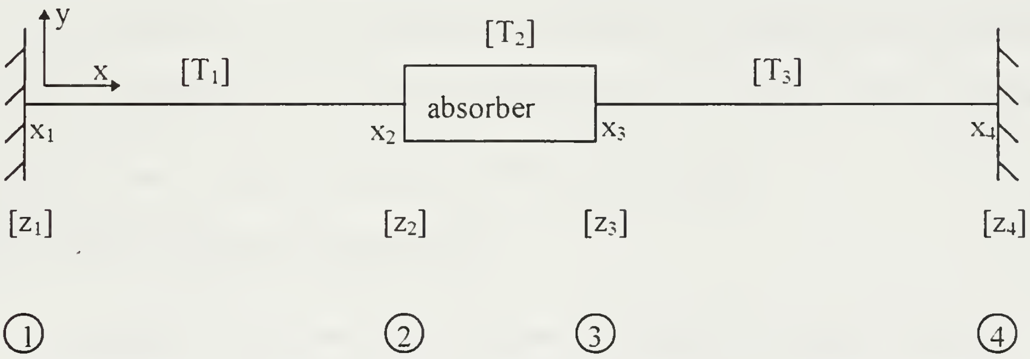
where $\overline{y}$ is the Fourier transform of cable transverse displacement and $\overline{y}'$ is the Fourier transform of cable slope. The product of cable tension T and $\overline{y}'$ is a value equal to the transverse force caused by tension at the location of the state vector. Thus, the cable system state vector contains a displacement value and a force value. If assumption 2 in section 2.1.1 were not valid and cable bending stiffness could not be neglected, the state vector would require four elements: displacement, slope, shear force and bending moment. The use of Fourier transforms is useful since the types of motion studied here will limited to free and forced harmonic vibrations.

Within a mechanical system, the state vector at any location can be obtained by multiplying the state vector at some other location in the system by the transfer matrix of the component(s) between the two locations. In a marine cable system, the transfer matrices are 2 by 2 matrices represented as follows:

$$[T]=\begin{bmatrix}t_{11} & t_{12}\\ t_{21} & t_{22}\end{bmatrix} \qquad (2\text{-}2)$$

The elements of the transfer matrices are derived using the equations of motion. Since they relate the Fourier transforms of displacement and force, the transfer matrix elements are functions of circular frequency, ω. The transfer matrix of a component is general in that it does not change when the component is installed in other systems. Cable system components modeled here using transfer matrices include cable segments, lumped and distributed masses and vibration absorbers.

Figure 2-1 illustrates how transfer matrices are used to relate the state vectors at individual locations in the cable system:



$$[z_2] = [T_1]\,[z_1] \qquad (2\text{-}3)$$

$$[z_3] = [T_1]\,[T_2]\,[z_1] \qquad (2\text{-}4)$$

$$[z_3] = [T_2]\,[z_2] \qquad (2\text{-}5)$$

$$[z(x)] = [T_1(x)]\,[z_1]\ ;\ x < x_2 \qquad (2\text{-}6)$$

Figure 2-1: Transfer Matrix Representation of Cable / Absorber System

The transfer matrix for the uniform cable section, $[T_1]$, comes directly from the solution to the linear string wave equation and is given by:

$$[T_1] = \begin{bmatrix} \cos(kl) & 1/kT\,\sin(kl) \\ -\,kT\,\sin(kl) & \cos(kl) \end{bmatrix} \qquad (2\text{-}7)$$

where: k = wave number = ω / c (m^{-1})

c = wave propagation speed = $(T / \rho)^{1/2}$ (m/s)

T = cable tension (N)

ρ = cable linear mass density (kg/m)

l = cable length from x_1 to x_2 (m)

For locations on the cable to the left of x_2, substitution of x for l in equation 2-7 will yield the transfer matrix $[T_1(x)]$ as in equation 2-6.

The transfer matrix for the absorber, $[T_2]$, is given in Appendix B and in the thesis by Li [5]. Appendix B also contains the transfer matrices for various vibration absorbers and discrete masses which may be located within a cable system.

2.1.3 The Overall Transfer Matrix

The product of the three transfer matrices, $[T_1]$ $[T_2]$ $[T_3]$, equals the overall transfer matrix of the system. For the system shown in figure 2-1, the boundaries are walls. For a wall, cable displacement would be zero and cable force would be unknown. Thus, the state vectors at the boundaries can be represented by:

$$[z] = \begin{bmatrix} 0 \\ T\bar{y}'(x) \end{bmatrix} \qquad (2\text{-}8)$$

The overall system transfer matrix can be used to relate the two boundary state vectors:

$$[T_{\mathrm{overall}}] = [T_1]\,[T_2]\,[T_3] \qquad (2\text{-}9)$$

$$[z_4] = [T_{\mathrm{overall}}]\,[z_1] \qquad (2\text{-}10)$$

$$\begin{bmatrix} 0 \\ T\bar{y}'(x_4) \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} 0 \\ T\bar{y}'(x_1) \end{bmatrix} \qquad (2\text{-}11)$$

The zero displacement at the boundaries results in:

$$(t_{11} \times 0) + \left(t_{12} \times T\bar{y}'(x_1)\right) = 0 \qquad (2\text{-}12)$$

$$\Rightarrow t_{12} = 0 \qquad (2\text{-}13)$$

This is the characteristic equation for the system. Since transfer matrix elements are functions of frequency, the roots of t_{12} of $[T_{overall}]$ are the natural frequencies of the system.

The mode shapes of the system can then be plotted by setting a unity force at the wall:

$$[z_1] = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad (2\text{-}14)$$

It follows that:

$$[z(x)] = [T_1(x)]\,[z_1] \;\; ; x \leq x_2 \qquad (2\text{-}15)$$

$$= [T_1]\,[T_2]\,[T_3(x)]\,[z_1]\; ; x \geq x_3 \qquad (2\text{-}16)$$

Note that $[T_3(x)]$ is expressed by substituting the quantity $(x - x_3)$ for l into equation 2-7. This coordinate transformation is required because transfer matrix expressions normally reference a coordinate system which is located on the component that the transfer matrix represents.

In general,

$$[z(x)] = [T(x)]\,[z_1] = \begin{bmatrix} t_{11}(x) & t_{12}(x) \\ t_{21}(x) & t_{22}(x) \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad (2\text{-}17)$$

where $[T(x)]$ is the transfer matrix product of the system components between the left wall and x.

The first element of [z(x)] is therefore:

$$\bar{y}(x) = t_{12}(x) \qquad (2\text{-}18)$$

Equation 2-18 specifies the displacement mode shape of the system. Note that $t_{12}(x)$ in equation 2-18 is an element of the transfer matrix between the left wall and an arbitrary point, x, and is not the same as t_{12} of $T_{overall}$ in equation 2-13.

In chapter 3, this approach will be applied to a specific marine cable / absorber system.

Chapter 3

Vibration Absorber Analysis

As mentioned in the introduction, the vibration absorber which will be the primary focus of this analysis is that shown below in figure 3-1:

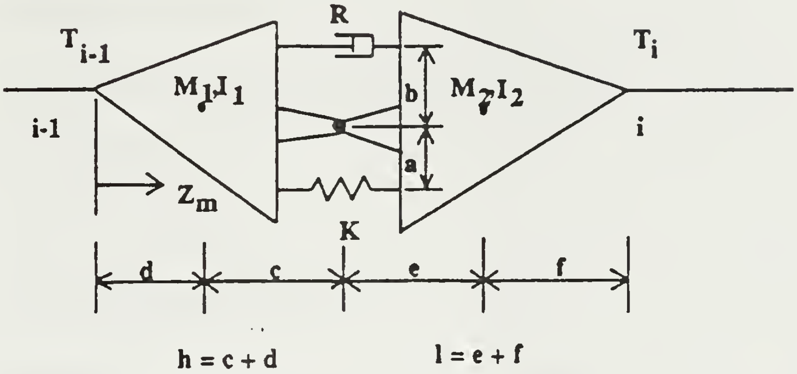


Figure 3-1: In Line Vibration Absorber

The transfer matrix expression for this absorber was derived by Li [5] and is quite lengthy, requiring a page and a half to state (see Appendix B). This is because the transfer matrix must account for both the translational and rotational motion of the masses, damping and stiffness. MATLAB was used to compute the values of this transfer matrix and was used for all other modeling and calculation in this thesis. Appendix A contains the MATLAB programs STRING1 and STRING2 which calculate the transfer matrices of the absorber and cable segments, respectively.

3.1 Modeling the Vibration Absorber in a Cable System

Since the nature of this analysis was numerical, a baseline absorber system of the type shown in figure 3-1 was modeled. System parameters such as cable length, cable diameter and absorber mass were set to values which could realistically be expected in a

marine system. Optimum absorber performance can be obtained by systematically varying these parameters. Baseline system parameters are as follows:

Total system length:	50.8 m
Cable tension, T:	5000 N
Cable density, ρ:	1 kg/m
Absorber link mass, M:	9 kg
Absorber link moment of inertia, I:	0.12 kg m^2
a and b dimensions:	0.03 m
c, d, e, f dimensions:	0.2 m
Total absorber length, L_{abs}:	0.8 m
Stiffness, K:	10 N/m
Damping, R:	varies N/(m/s)

Note that the stiffness, K, is relatively insignificant. Cable tension is the dominant factor in providing a restoring force for the absorber.

Absorber analysis consisted of the following steps:

1) Obtain natural frequencies

2) Determine damping ratio

3) Determine displacement mode shapes

4) Vary system parameters to optimize damping ratio

This process is detailed in the following sections.

3.2 The Cable / Absorber System With no Damping

It is valuable to first analyze the system and demonstrate the analysis process with the damping value of the absorber set to zero. This keeps all calculations in the real domain and provides for a better understanding of the introduction of damping.

3.2.1 System Natural Frequencies

Chapter 2 detailed the process for determining the characteristic equation and natural frequencies using transfer matrices. The overall transfer matrix of the system, $[T_{overall}]$, is computed as follows:

$$[T_{overall}] = [T_1]\ [T_2]\ [T_3] \qquad (3\text{-}1)$$

where:

$[T_1]$ and $[T_3]$ are the transfer matrices of the left and right cable segments as in equation 2-7

$[T_2]$ is the transfer matrix of the absorber as in Appendix B

Due to the fixed wall boundary conditions, the system characteristic equation is

$$t_{12}=0 \qquad (3\text{-}2)$$

where t_{12} is the first row, second column element of $T_{overall}$.

A MATLAB script file, STRING4, was used to generate figure 3-2 on the following page, a plot of t_{12} versus frequency, ω. The zeroes of t_{12} are the natural frequencies of the system.

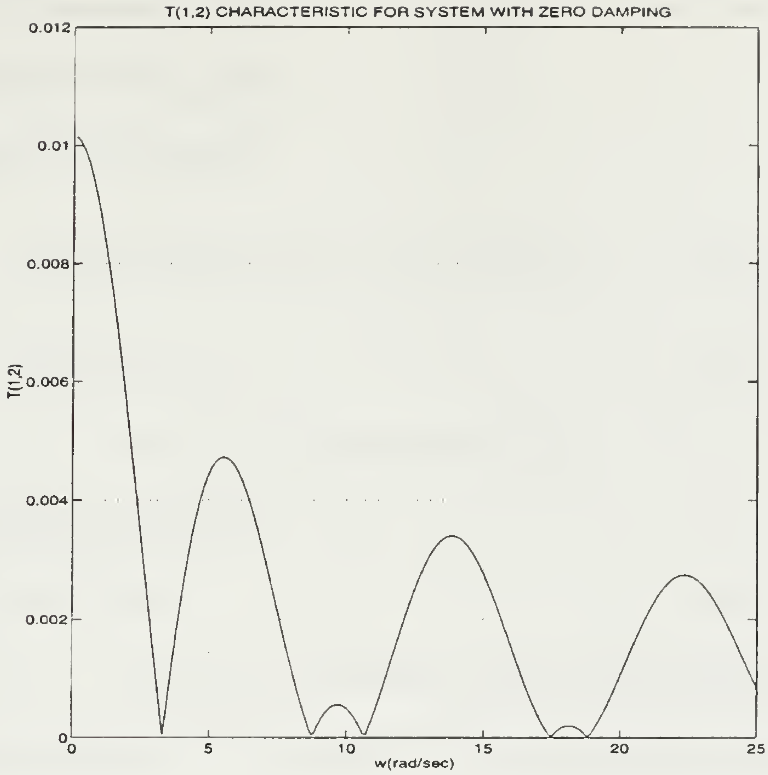


Figure 3-2: t_{12} Characteristic of Undamped System

3.2.2 System Mode Shapes

Once the system natural frequencies are known, transfer matrix multiplication can be used to determine the displacement and force mode shapes. This is accomplished by a MATLAB program called STRING6 which takes the system frequency as an input. The state vector at the left wall is set as follows

$$z_{left} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3-3}$$

where the first element indicates the zero displacement at the wall and the second element is set at one. This results in a mode shape which is normalized to a unity transverse force

at the system boundaries. The displacement mode shape is calculated along the length of the system as described in Chapter 2:

from left wall to absorber:

$$[z(x)] = [T_1(x)][z_{left}] = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \bar{y}(x) \\ T\bar{y}'(x) \end{bmatrix} \qquad (3\text{-}4)$$

$$\bar{y}(x) = t_{12}(x)\,, \qquad (3\text{-}5)$$

where $t_{12}(x)$ is an element of $[T_1(x)]$

from absorber to right wall:

$$\bar{y}(x) = t_{12}(x)\,, \qquad (3\text{-}6)$$

where $t_{12}(x)$ is an element of the matrix product $[T_1]\ [T_2]\ [T_3(x)]$.

Similarly, the transverse force mode shape would equal the t_{22} element of the transfer matrix as a function of distance from the left wall. The resulting displacement mode shapes for the first three vibration modes are shown in figure 3-3 on the following page.

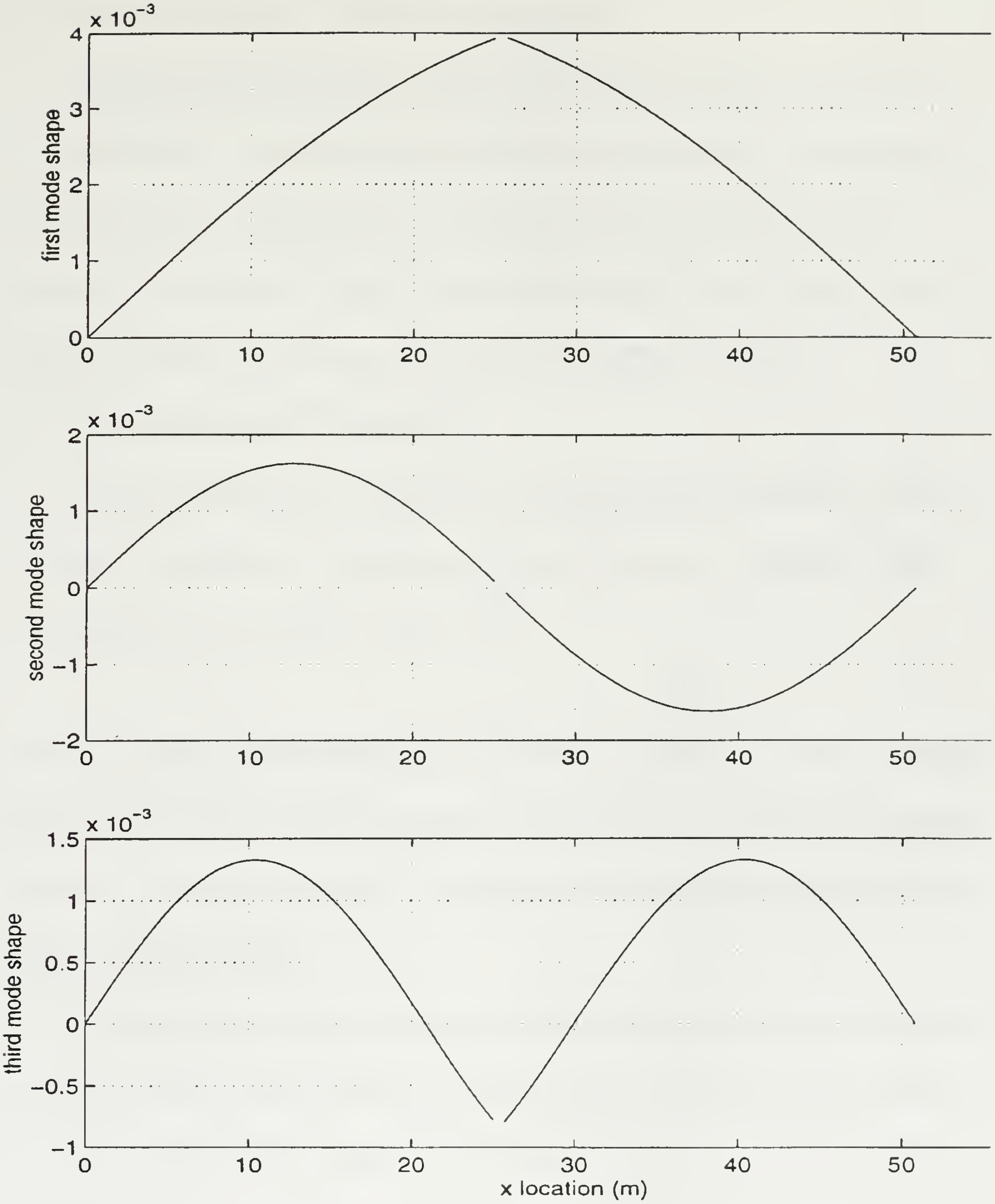


Figure 3-3: Undamped System Displacement Mode Shapes

3.3 Introduction of Absorber Damping

The introduction of absorber damping takes the transfer matrix calculations into the complex domain. In addition to natural frequency and mode shape, damping ratio can now be determined. Damping ratio is the parameter used in determining absorber effectiveness. Damping ratio is used to compute the reduced damping in figure 1-2 and to determine if sufficient damping exists to reduce vortex-induced vibrations.

3.3.1 System Natural Frequencies

The same approach which was used to compute undamped natural frequencies in section 3.2.1 is used here to compute damped natural frequencies. As shown in section 3.2.1, the characteristic equation of the system is:

$$t_{12}=0 \qquad (3\text{-}7)$$

where t_{12} is an element of the overall system transfer matrix. Figure 3-4 on the following page shows the magnitude of t_{12} as a function of frequency, ω. Note that when damping is present, t_{12} is a complex quantity. The figure also shows t_{12} for the undamped system in the same frequency range

The damped curve is quite different in appearance from the undamped case. For alternating modes, the damped value of t_{12} fails to reach a zero for *real* values of natural frequencies. This behavior is significant and will later be shown to have a very physical explanation.

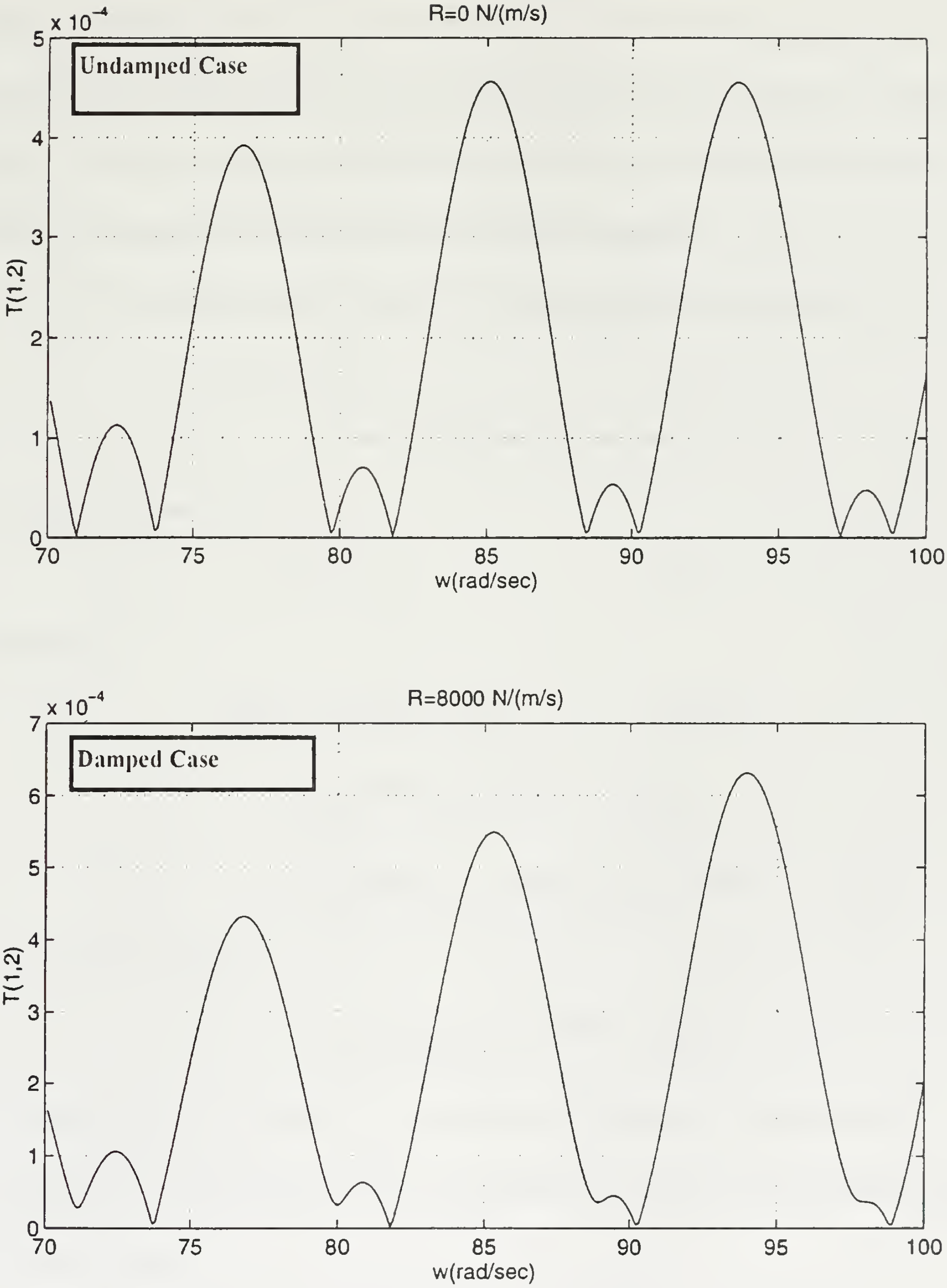


Figure 3-4: Magnitude(t_{12}) Versus Frequency for Damped vs. Undamped System

In order to obtain all the zeroes of t_{12}, complex values of frequency must be considered. This requires the use of a computer complex root solver, MULLER (Appendix A), developed by Conte and de Boor [14] and converted into MATLAB format by Wilson and Roberts. Before proceeding further with the results of the complex root search, the meaning of a complex natural frequency must first be discussed.

The harmonic motion at a point in the cable system can be described by:

$$y\,(x) = Ae^{i\omega t}, \qquad (3\text{-}8)$$

where A is displacement amplitude.

For imaginary natural frequencies:

$$\omega = \omega_{real} + i\omega_{imag} \qquad (3\text{-}9)$$

Substituting:

$$y(x) = Ae^{i(\omega real + i\omega imag)t} \qquad (3\text{-}10)$$

$$= A\,e^{i(\omega real)t}\,e^{-(\omega imag)t} \qquad (3\text{-}11)$$

$$= A[\cos(\omega_{real}t) + i\sin(\omega_{real}t)]\,e^{-(\omega imag)t} \qquad (3\text{-}12)$$

The real part of which is:

$$= A\,\cos(\omega_{real}t)\,e^{-(\omega imag)t} \qquad (3\text{-}13)$$

So, we see that for free vibrations the damped system vibrates at ω_{real} with a decay envelope $e^{-(\omega imag)t}$. The ratio of the real and imaginary parts of the natural frequency is the modal damping ratio:

$$\zeta = \omega_{imag} / \omega_{real} \qquad (3\text{-}14)$$

Now that the significance of a complex natural frequency is understood, we can return to the analysis of the cable / absorber system. The complex root solver, MULLER,

is iterative and requires a starting guess for the complex frequency root. The undamped natural frequencies were used as initial guesses when running MULLER. With the assistance of Dr. Rama Rao, MIT, a routine called MULRUN (Appendix A) was developed which successively runs MULLER for many values of damping constant to obtain an output which demonstrates the optimum value of damping. Figure 3-5 below shows the output of MULRUN for the first vibration mode of the baseline system.

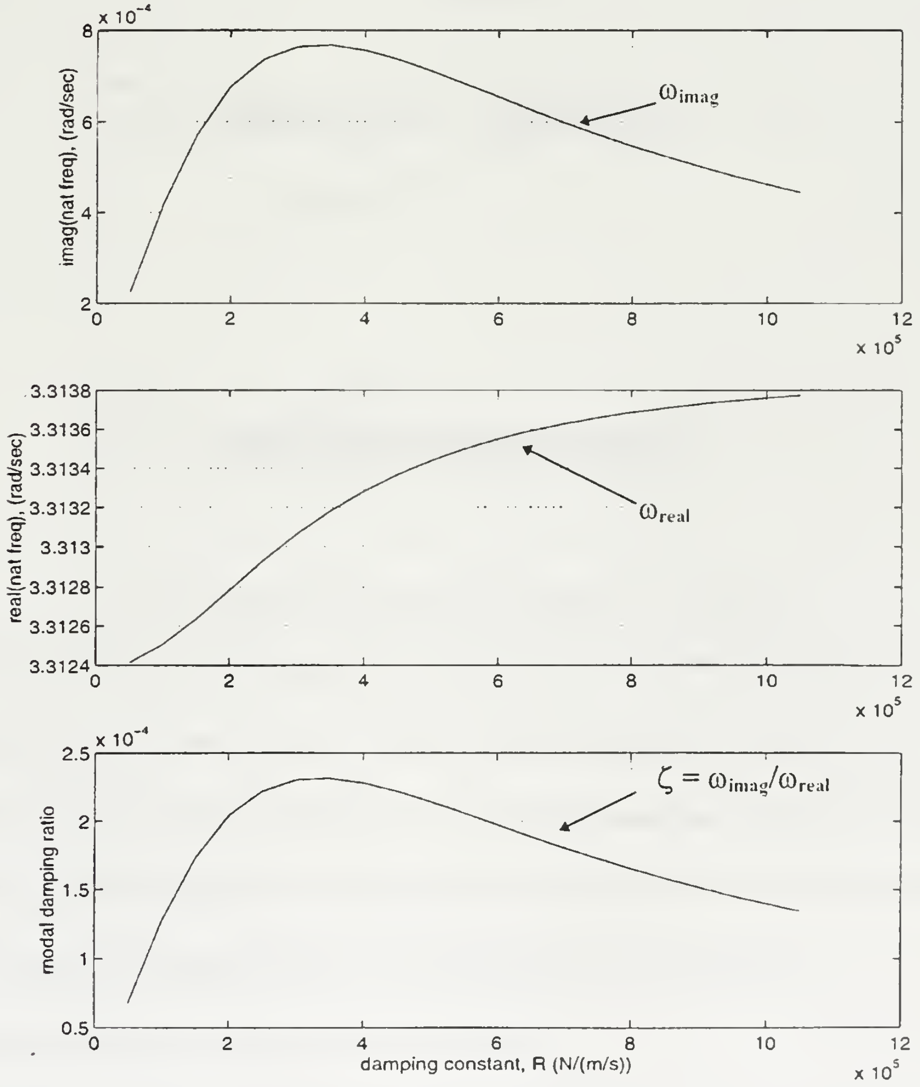


Figure 3-5: Demonstration of Optimum Damping Value

The MULRUN output shows that as damping constant, R, is varied, there is an intermediate damping value which optimizes (maximizes) damping ratio. It was theorized that the reason damping ratio decreases at high values of damping constant is that relative motion between the absorber masses decreases above the optimum value. This was confirmed by creating a transfer matrix model of a uniform bar (see Appendix B) in place of the vibration absorber in the baseline system as shown in figure 3-6:

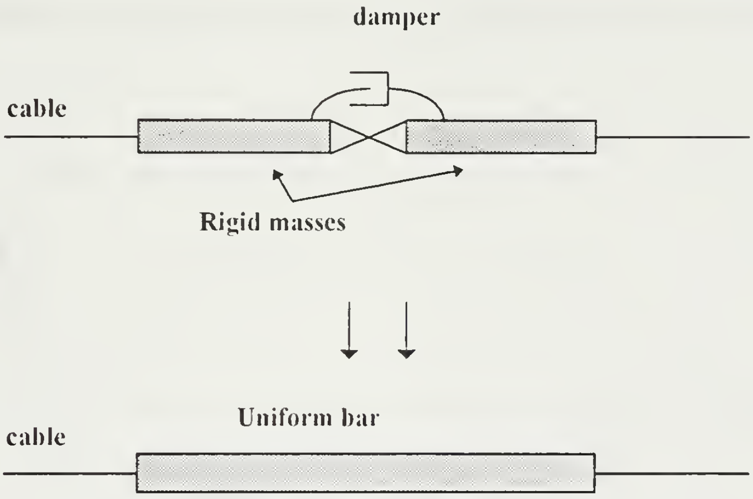


Figure 3-6: Representing High Stiffness Absorber as Uniform Bar

The uniform bar had the same dimensions and mass as those of the absorber. It was shown that the cable system with the uniform bar at its center had the same natural frequency and mode shape as the absorber with very high damping constant. So at very high damping, relative rotational motion between the absorber links is restricted and the absorber is unable to dissipate energy. The absorber acts as a uniform bar.

3.3.2 System Mode Shapes With Damping

Once the system natural frequencies are known, transfer matrix multiplication can be used to determine the displacement and force mode shapes using the same approach as

for the undamped case. This is accomplished by a MATLAB program called STRING6 which takes system frequency as an input. For the damped case, complex natural frequencies are input into STRING6. The state vector at the left wall is again:

$$[z_{left}] = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad (3\text{-}15)$$

The displacement mode shape is calculated along the length of the system as in section 3.2.2. The resulting mode shapes for the first three vibration modes are shown in figure 3-7.

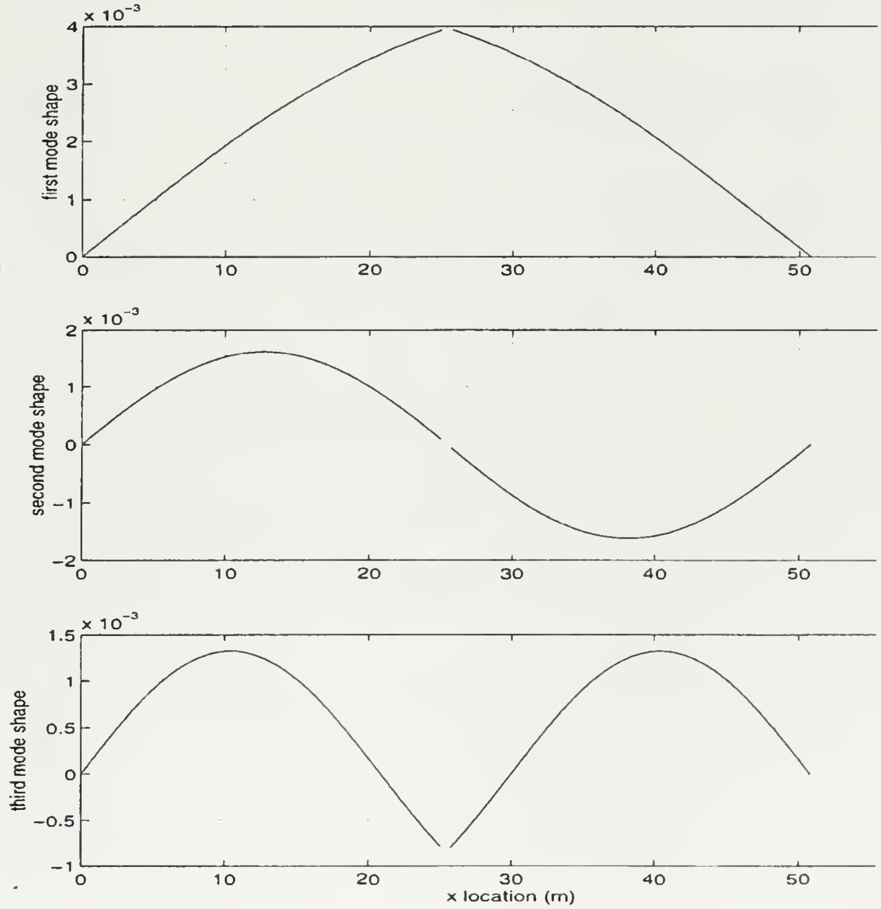


Figure 3-7: Damped System Displacement Mode Shapes

Inspection of these mode shapes reveals that for the even mode, the absorber is located at a modal antinode. There is no relative motion between the absorber links. The absorber simply rotates about its center as if it were a uniform bar. This is confirmed further by using MULRUN for the same modes to determine optimum damping ratio. In figure 3-8 below, it is seen that for the even vibration mode, damping ratio is insignificant even though damping is present. Relative motion between the absorber links is required if the absorber is to dissipate energy.

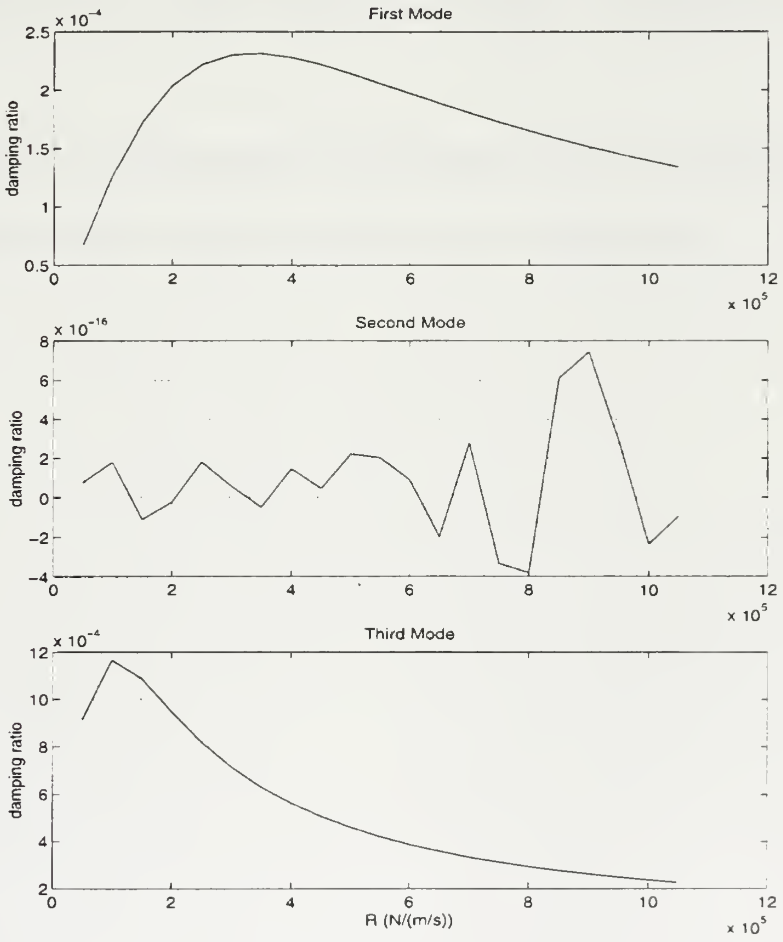


Figure 3-8: Damping Effect in First Three Vibration Modes

So, from initial analysis the following observations are made:

1) An optimum value of damping constant is one which adds sufficient damping while not overly restricting motion between the two absorber links.

2) Absorber effectiveness is very dependent on absorber location. Absorbers located at modal antinodes provide the most energy dissipation.

These observations are consistent with the behavior of more conventional types of vibration absorbers, such as those installed in discrete (vice continuous) mechanical systems.

In chapter 4, parameters such as absorber mass, absorber length and number of absorbers will be varied to study their effect on absorber performance.

Chapter 4

Parameters Affecting Absorber Performance

In previous chapters, it has been demonstrated how natural frequencies, mode shapes and damping ratios can be obtained for a vibration absorber in a marine cable system. It was shown how the damping constant, R, can be varied to arrive at an optimum damping ratio. The tools are now in place to study the effects of other parametric variations on absorber performance. Improvements in absorber performance will be sought by varying the following:

1) Absorber length
2) Absorber mass
3) Number of absorbers installed in a single cable system
4) Absorber location

4.1 Required Damping

Before proceeding with further attempts to optimize damping, it would be helpful to know what value of damping is required in the system to reduce vortex induced vibrations. Recall from figure 1-2 that there is a value of reduced damping below which structural damping provides no assistance in reducing vortex induced vibrations. Structures with reduced damping in this plateau range will have displacement amplitudes which are restricted by limit-cycle reductions to the lift forces. Thus, there is no gain in adding small amounts of damping which will not raise reduced damping above that value required to reduce the response to less than that resulting from limit cycle behavior. From figure 1-2, this threshold value of reduced damping is as follows:

$$S_G \geq 0.2$$

where:

$$S_G = 8\,\pi^2\,S_t^2\,\zeta\,(\,m_s\,/\,\rho_f\,D^2) \qquad (4\text{-}1)$$

For the baseline system in water (see section 3.1), this results in a threshold (required) value of damping ratio:

$$\zeta \geq .04 \qquad (4\text{-}2)$$

The highest value of damping ratio achieved thus far (see figure 3-8) is approximately $\zeta = .0012$. So, we see that if the absorber being studied is to be of use in reducing vortex induced vibrations in water, further optimization is required.

4.2 Variation of System Parameters

4.2.1 Absorber Length

In figure 3-7, mode shapes for the first three modes were shown. These low modes have very long wavelengths, resulting in the situation shown in figure 4-1(a) on the following page. When absorber length is small compared to wavelength, relative angular motion between the absorber links is very small, even in odd numbered vibration modes. Since energy dissipation relies on this relative motion, short absorber links result in poor absorber performance. As absorber length is increased, relative motion between the absorber links is increased, and energy dissipation increases as shown in figure 4-1(b). A key dimensionless parameter of use here would be the ratio of absorber length to the half wave length of a given mode. This parameter is designated as the length ratio:

$$LR = 2\,L_{absorber}\,/\,\lambda \qquad (4\text{–}3)$$

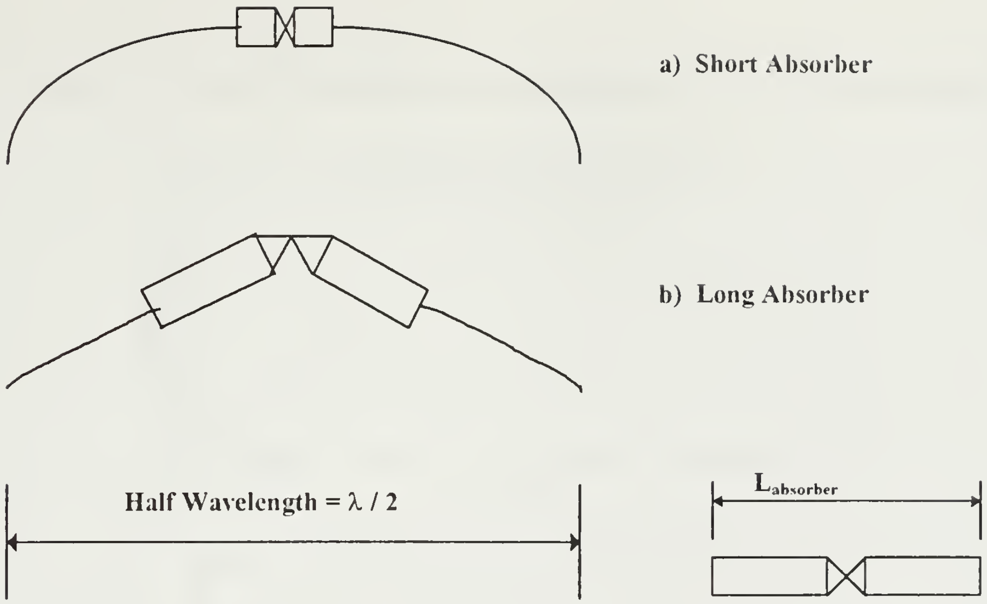


Figure 4-1: Effect of Absorber Length on Damper Motion

Variation of absorber length was accomplished in an indirect way. Instead of varying the absorber link length, the effect of absorber length was studied by varying λ, or simply by comparing the damping ratios of different modes of the baseline system. Starting with mode 1, optimum damping values were determined for each mode. As discussed in section 3.3.2, even numbered vibration modes exhibited no damping effects. The best absorber performance (as determined by maximum damping ratio) was achieved in the 29th mode.

In this mode:

$$\lambda = 3.50 \text{ m}$$

$$L_{absorber} = .8 \text{ m}$$

$$LR = .457$$

In modes beyond mode 29, higher length ratio resulted in more restricted absorber motion and reduced damping ratios.

Figure 4-2 shows the damping value optimization and mode shape for mode 29.

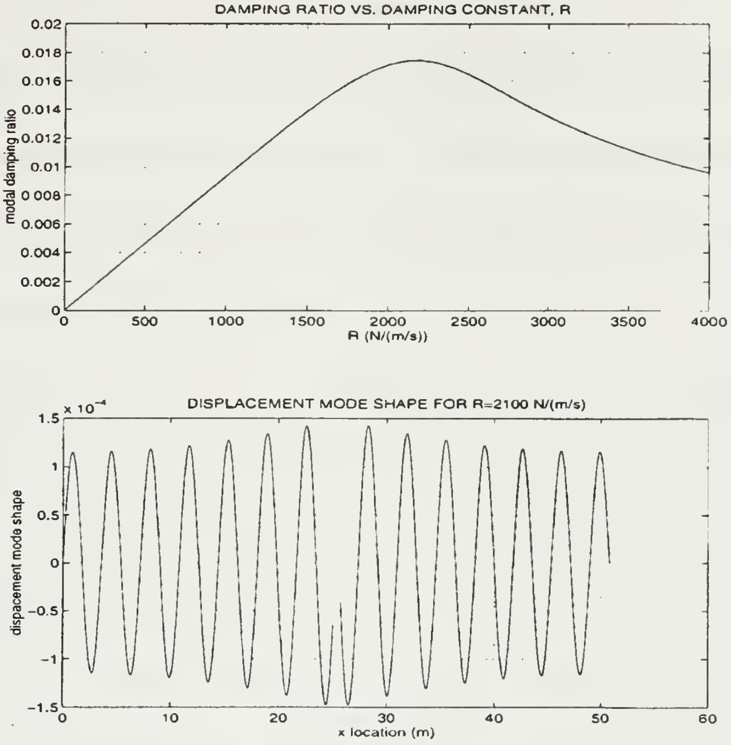


Figure 4-2: Damping Optimization and Mode Shape for Mode 29

It can be seen that with an optimum value of damping constant, damping ratio is approximately 1.7%. This falls well short of the value of 4% required to reduce vortex induced vibration amplitude in water.

4.2.2 Absorber Mass

In further efforts to increase the amount of damping which the absorber adds to the system, the mass of the absorber links was varied. A useful dimensionless parameter for this analysis was the mass ratio:

$$MR = m_{absorber} / (L_{absorber}\, \rho_{cable}) \qquad (4\text{-}4)$$

This is the ratio of the absorber mass to the mass of the cable segment displaced by the absorber.

The variation of mass ratio was performed for the 29th mode of vibration of the cable system. This mode was selected because it this was the mode for which optimum absorber length was shown . Thus the result of this variation represents an attempted optimization of both absorber mass and length. There is no guarantee that optimum length ratio remains at 0.457 when mass ratio is varied. Figure 4-3 shows how optimum damping ratio varies with absorber mass:

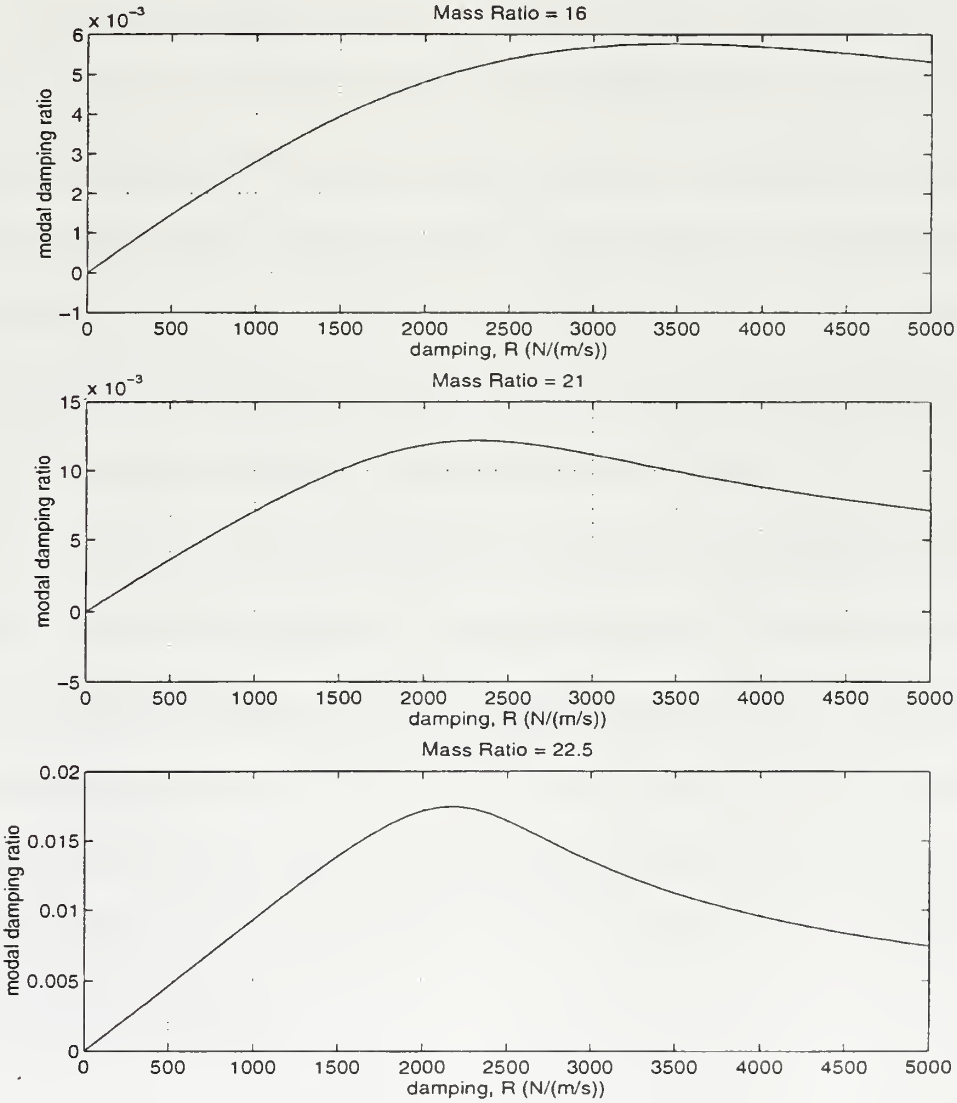


Figure 4-3: Affect of Mass Ratio on Mode 29 Damping

Figure 4-3 shows that increasing absorber mass ratio tends to increase optimum damping ratio. Also, lower values of damping constant are required to achieve the optimum damping value when a higher mass ratio is used. This figure demonstrates that the baseline system (MR=22.5) has the optimum mass ratio because it results in the highest system damping ratio. When mass ratio was increased beyond 22.5, the system shifted to an even numbered vibration mode where damping is insignificant.

If this type of analysis were to be performed on a real marine system, one would have to exercise care in changing absorber mass. Changing absorber mass changes the mass of the vibrating system and thus changes system natural frequencies. The marine system would be designed with some range of current velocity and corresponding vortex shedding frequencies in mind. If one were not careful, one could change system mass such that the absorber was optimized at some frequency outside the range of concern.

4.2.3 Behavior at Realistic Values of Absorber Mass

Admittedly, a mass ratio of MR=22.5 corresponds to a very large absorber mass, equaling 36% of the total cable mass. Absorbers of this extreme size were considered only in an attempt to achieve the required damping level of section 4.1. A more typical absorber mass might be no more than 10% of the cable mass. This would correspond to a mass ratio of MR=6.25. In this section, an absorber of this size was analyzed over many vibration modes and corresponding length ratios. The results are shown in figure 4-4 and 4-5 on the following page.

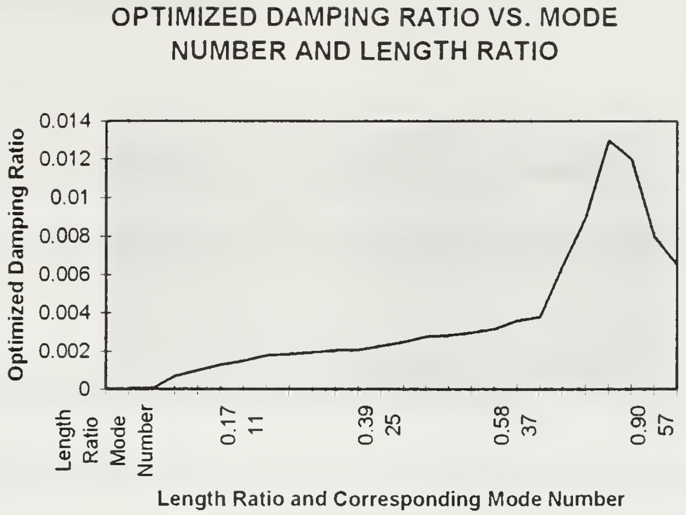


Figure 4-4: Optimized Damping Ratio vs. Mode Number and Length Ratio for Mass Ratio = 6.25

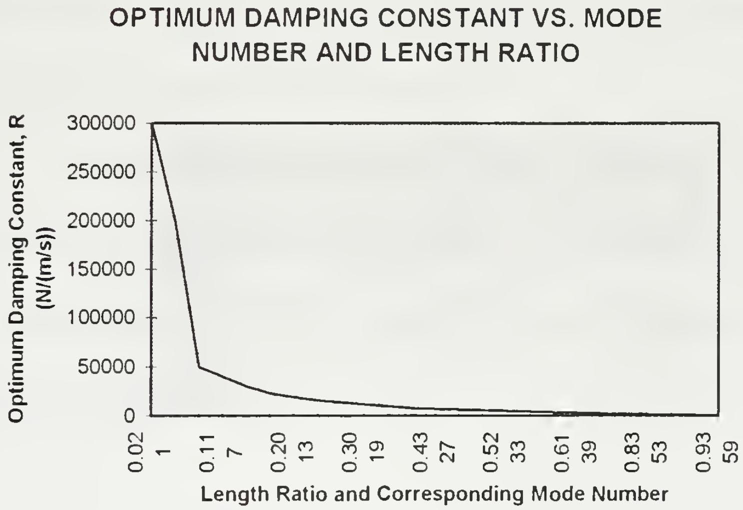


Figure 4-5: Optimized Damping Constant vs. Mode Number and Length Ratio for Mass Ratio = 6.25

Figures 4-4 shows that an optimum damping ratio is achieved in mode 53 with a length ratio of LR=0.83. Figure 4-5 shows how required damping decreases dramatically with increasing length ratio. This behavior was discussed in section 4.2.1.

4.3 Number of Absorbers

Up to this point, variation of system parameters has not resulted in achieving the required damping ratio of Section 4.1. It would be very convenient if one could multiplicatively increase damping ratio by simply increasing the number of absorbers installed.

This addition of absorbers was performed for mode 3. A transfer matrix model was constructed for a cable system with vibration absorbers at its mode 3 antinodes. An absorber mass ratio of MR=6.25 was used. All other system parameters were those of the section 3.1 baseline system. The MATLAB programs which model this system are located in directory MULTABS in Appendix A. These programs are functionally identical to those used to evaluate the system with one absorber.

Figure 4-4 on the following page shows the mode 3 mode shapes for the system with one absorber and the system with three absorbers. For both cases, a mass ratio of MR=6.25 and a length ratio of LR=.047 was used. The maximum damping value achieved in each case is indicated.

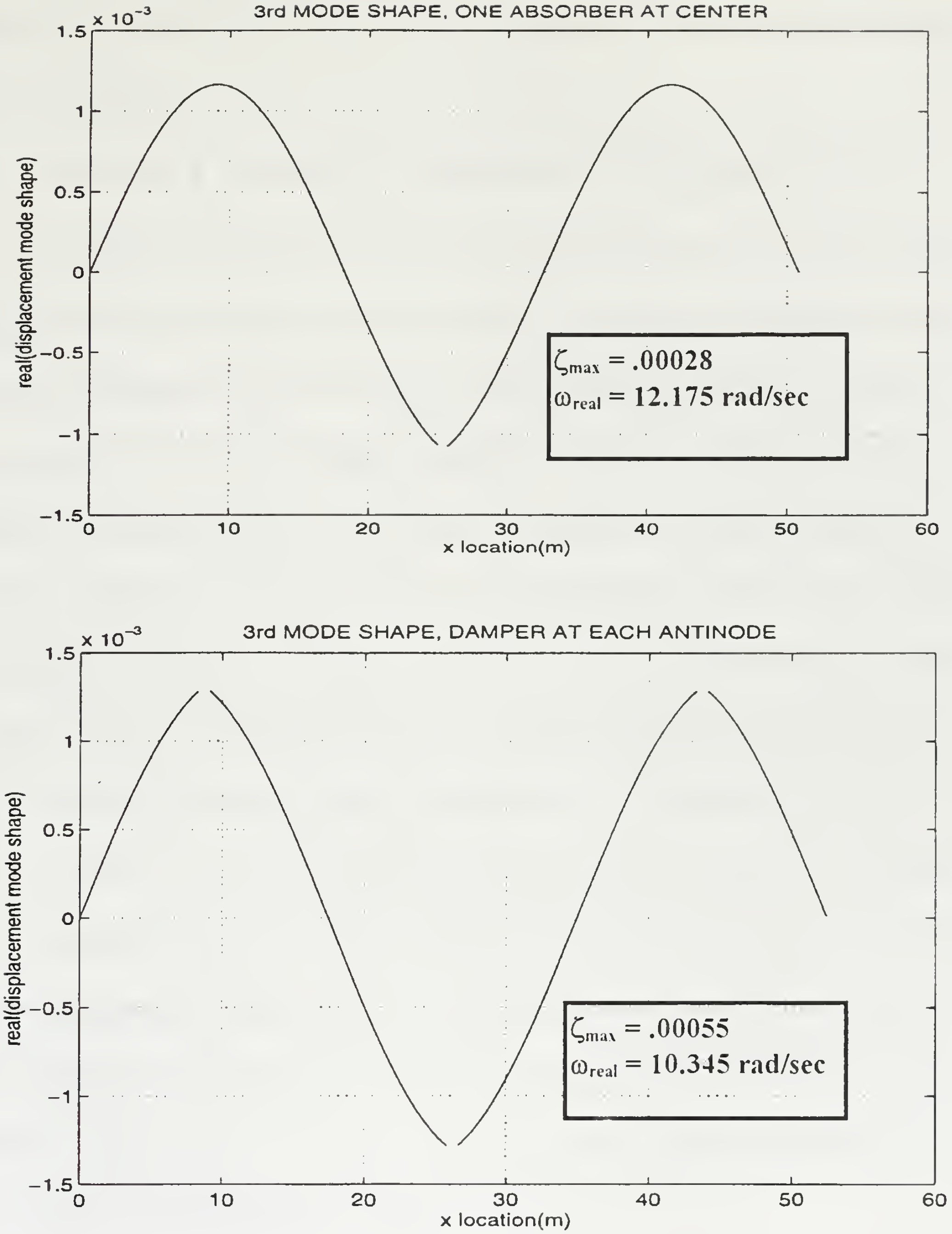


Figure 4-6: Comparison of Single Absorber to Multiple Absorbers

Inspection of figure 4-6 reveals that optimized damping ratio roughly doubled when the number of absorbers was tripled. Thus, damping ratio is not multiplicative with the number of absorbers installed in the system. One should also observe from figure 4-4

that mode 3 natural frequency decreases considerably with the additional mass loading of the extra absorbers.

4.4 Absorber Location for Operation in Multiple Modes

In previous sections, absorber performance has been looked at one mode at a time. Parameters such as damping constant and absorber mass have been varied and the effect on modal damping ratio has been observed. This has improved understanding of the system but it is not the way in which a marine vibration absorber system would be designed. In a system subject to vortex induced vibrations, some range of potential currents would exist. This range of currents would be directly correlated to an excitation bandwidth by the Strouhal relationship of equation 1-1. So, an absorber design problem would be concerned with multiple modes. This section will illustrate how the proper location of a single absorber can optimize performance in multiple vibration modes.

In section 3.3.2, it was observed that the most effective location for an absorber was at a modal antinode. For mode 3, these locations would be L/6, L/2, and 5L/6. Now suppose we wanted to design an absorber system which would be effective for modes 3 and 4. The mode 4 antinodes are at L/8, 3L/8, 5L/8. The two modes share no common antinodes. In fact, the L/2 antinode for mode 3 is a mode 4 node. So, an absorber located at L/4 offers no absorber effect for mode 4.

A compromise was arrived at by constructing a model of a system with an absorber at 7L/16. This location is half way between the 3L/8 mode 4 antinode and the L/2 mode 3 antinode. The absorber used had a mass ratio of MR=6.25. The MATLAB programs for modeling and analyzing this system are located in directory OFFCENTER in

Appendix A. Table 4-1 below compares absorber effectiveness for the compromise absorber location versus absorber locations optimized for a single mode:

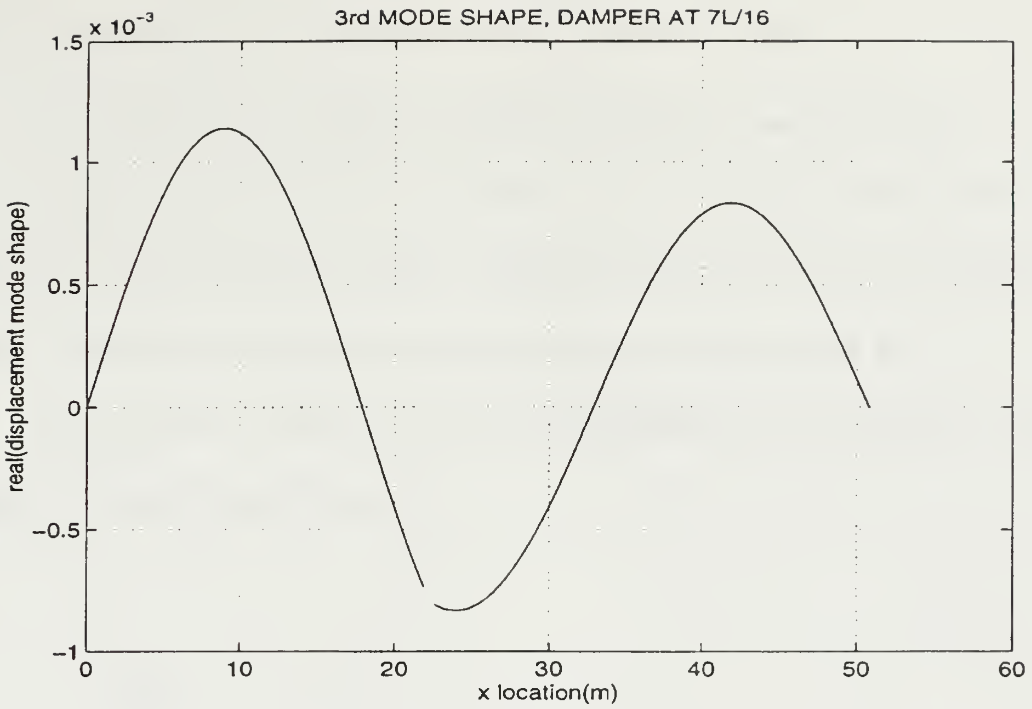
	Absorber Located for Mode 3 Effectiveness	Absorber Located for Dual Mode Effectiveness
Mode 3 Damping Ratio	.00028	.00023
Mode 4 Damping Ratio	0	.00023

Table 4-1: Dual Mode Absorber Optimization

The results indicate that relocation of the absorber causes some loss of mode 3 damping ratio while raising mode 4 damping ratio from zero to a value equal to that of mode 3. This is the type of tradeoff would have to be performed in the design of a real marine cable absorber system.

Figure 4-7 on the following page shows the mode 3 and 4 mode shapes with the absorber located at 7L/16. This figure demonstrates the utility of the STRING6 mode plotter in visualizing unorthodox mode shapes.

So, absorbers can be located for effectiveness in both even and odd numbered modes. In choosing absorber location, the designer of a marine cable system would have to consider the excitation bandwidth and vibration modes within that bandwidth. If a single absorber location could not provide energy dissipation in all possible excitation modes, then multiple absorbers would need to be installed.



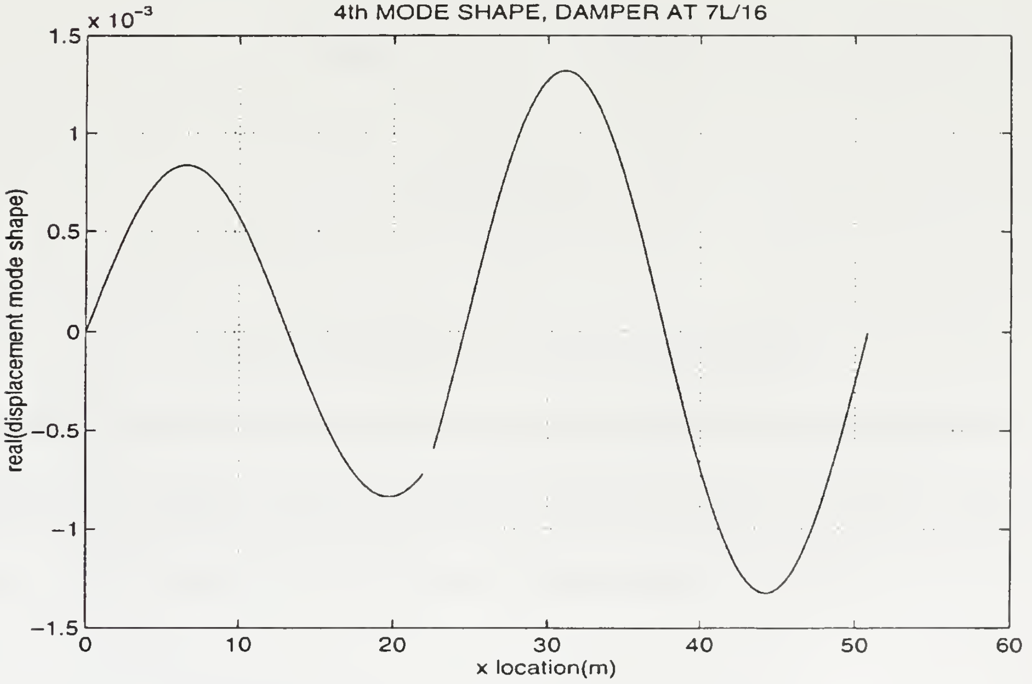


Figure 4-7: Mode 3 and 4 Shapes with Absorber at 7L/16

4.5 Consideration of a Different Absorber Type

Up to this point in the analysis, considerable insight has been gained in the behavior of the hinged, in-line, vibration absorber in the marine cable system. However, this type of absorber has not been able to produce the necessary damping ratio required to reduce vortex induced vibrations in a cable system in water. (See section 4.1).

In this section, a simpler and more conventional absorber will be analyzed. This absorber is shown in figure 4-8 below:

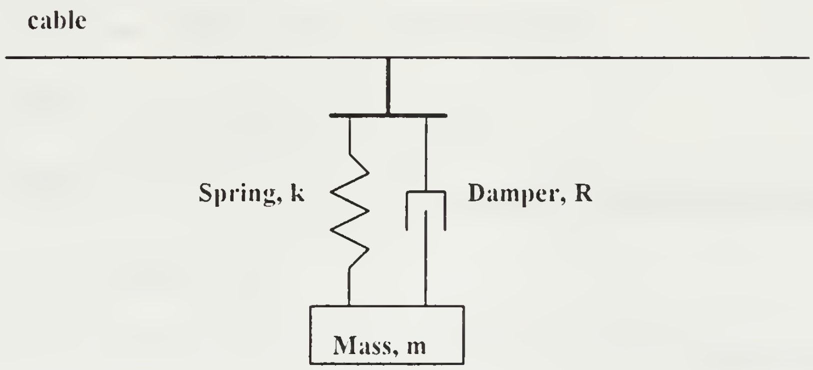


Figure 4-8: Hanging-Mass-Spring Dashpot Absorber

The absorber is of the conventional mass-spring-dashpot type. It is usually associated with discrete mechanical systems but could also be used to dissipate energy in a cable system. Li and Vandiver [10] derived the following transfer matrix for this absorber:

$$[T] = \begin{bmatrix} 1 & 0 \\ -\dfrac{\omega^2 m\,(j\omega R + k)}{k - \omega^2 m + j\omega R} & 1 \end{bmatrix} \qquad (4\text{-}5)$$

A numerical model of a cable system using this absorber was constructed in MATLAB. The cable itself is identical to that of the baseline system of section 3.1. However, the absorber of figure 4-8 is located at the center of the cable in place of the in-line bending type absorber. The programs comprising this model are located in directory HANG in Appendix A.

Den Hartog [1] formulated a method for choosing optimum absorber mass and stiffness for this type of absorber. His formulation was for discrete systems but modal analysis of the continuous cable system allows Den Hartog's method to be applied in an approximate way. The approach uses the following parameters:

$\mu = m / M$ = mass ratio = absorber mass/ main mass

$\omega = \sqrt{k / m}$ = natural frequency of absorber

$\Omega = \sqrt{K / M}$ = natural frequency of main system (cable without absorber)

$f = \omega / \Omega$ = frequency ratio

Den Hartog determined that the correct tuning of the mass spring dashpot absorber results from the frequency ratio being set as follows:

$$f = 1 / (1 + \mu) \qquad (4\text{-}6)$$

This equation was used to set absorber stiffness. To do this, an example mass ratio of $\mu = 0.2$ was selected. The main mass was set at the modal mass of the cable which equals one half of total cable mass. Ω was set at the cable first mode natural frequency. Equation 4-6 then allowed computation of the absorber stiffness, k.

The following values were used in the analysis of the vibration absorber for the first vibration mode of the cable system:

m = 5 kg = absorber mass

M = 25 kg = cable modal mass = ½ of total cable mass

$\mu = m / M = 0.2$

Ω = 4.44 rad/sec = cable first mode natural frequency without absorber

Using equation 4-6:

$f = 0.833 = \omega / \Omega$

$\omega = 3.70 \text{ rad/sec} = \sqrt{\dfrac{k}{m}}$

k = 68.5 N/m, the correct stiffness for optimum tuning

Once stiffness and mass were set, MULRUN was used to determine the optimum damping constant for the absorber designed in the procedure above. Figure 4-9 below shows the resulting damping ratio performance and first mode shape:

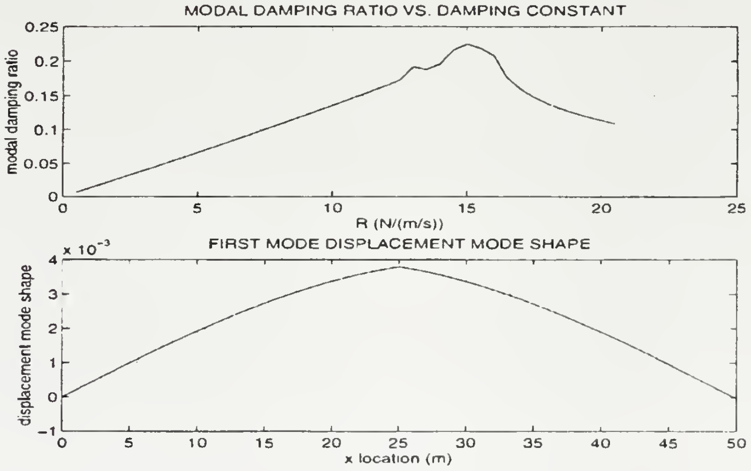


Figure 4-9: DamperOptimization for First Mode with Hanging Mass Spring Dashpot Absorber

It can be seen that optimum damping ratio is approximately 0.22, far higher than any damping ratio achieved with the in line absorber. This damping ratio is also above the critical value of $\zeta = .04$ necessary to reduce vortex induced vibration amplitude in water. This result also provides a reasonable reconciliation of results between the Den Hartog method and the transfer matrix method used in this thesis. Den Hartog states that the optimum damping constant for the mass spring dashpot absorber can be determined by the following formula:

$$\frac{R_{optimum}}{2\,\Omega\,m} \cong [\,3\,\mu\,/\,(8\,(1+\mu)^{3})]^{1/2} \qquad (4\text{–}7)$$

For a mass ratio of $\mu = 0.2$, this results in an $R_{optimum}$ of 9.25 N/(m/s). This corresponds roughly with the optimum damping constant of 15 N/(m/s) shown in figure 4-7.

The hanging mass-spring-dashpot absorber represents a large improvement in absorber effectiveness over the bending type absorber discussed elsewhere in this thesis. This absorber has some possibility for use in marine cable systems and further analysis should be conducted.

Chapter 5

Conclusion

In this thesis, the transfer matrix method was used to construct various numerical models of marine cable systems. The performance of various vibration absorber configurations was studied. The following are the main findings of this research:

1) Introduction of damping in the transfer matrix model causes the system characteristic equation to have complex roots. The use of a computer complex root solver was introduced and was shown to be useful in calculating system natural frequencies and damping ratios.

2) Upon understanding the significance of complex natural frequencies, computer programs using transfer matrices were created to plot system mode shapes. The ability to visualize mode shapes was very important in understanding system behavior.

3) The use of an in-line bending type vibration absorber was studied in some depth. A method was devised for tuning such an absorber to achieve maximum damping within a marine cable system. Variation of parameters such as absorber mass, length and number of absorbers was performed to study the effect of these variations on absorber performance. Even after optimization, the in-line absorber was unable to demonstrate sufficient damping to reduce vortex-induced vibrations in water.

4) The use of a hanging mass-spring-dashpot absorber was also studied for use in a marine cable system. This absorber was shown to have the ability to create sufficient damping to reduce vortex induced vibrations in water.

The main focus of this thesis was to study the performance of the in-line bending type absorber in a finite length marine cable system. Although it was shown that this absorber provides insignificant damping to reduce vortex-induced vibrations in water, considerable insight was gained in this thesis. A working model was created which has the ability to calculate system natural frequencies, mode shapes and damping ratios. Additionally, it should be noted that the in-line absorber could be useful in reducing vortex induced vibrations in air. The studies in this thesis would be useful in optimizing the absorber for this application.

Areas for potential research related to this thesis include:

1) Use of the transfer matrix models in this thesis to study other marine cable system characteristics, such as the effects of distributed masses within the system. Li [5] performed considerable work in this area but the ability to visualize mode shapes would provide a good supplement his work.

2) Extension of the analysis procedure in this thesis to other systems which can be represented by transfer matrices.

3) Further study of the mass-spring-daspot absorber for use in marine cable systems.

Bibliography

1. Den Hartog, J. P. (1955). Mechanical Vibrations. 4th edition. McGraw-Hill Book Company, New York.

2. Pestel, E. C. And Leckie, F. A. (1963). Matrix Methods in Elastomechanics. McGraw-Hill Book Company, New York.

3. Faltinsen, O. M. (1990). Sea Loads on Ships and Offshore Structures. Cambridge University Press, Cambridge, UK.

4. Fertis, D. G. (1995). Mechanical and Structural Vibrations. John Wiley & Sons, New York.

5. Li, Li (1993). The Dynamics of Strings with Rigid Lumps. Massachusetts Institute of Technology, Cambridge, MA.

6. Rao, Rama (1991). Acoustic Transmission Through Fluid Filled Pipes in Boreholes. Massachusetts Institute of Technology, Cambridge, MA.

7. Butler, E. L. (1996). Design and Testing of Vibration Absorbing Oceanographic Cable Terminations. Massachusetts Institute of Technology, Cambridge, MA.

8. Vandiver, J. K. (1993). "Dimensionless Parameters Important to the Prediction of Vortex Induced Vibration of Long, Flexible Cylinders in Ocean Currents". Journal of Fluids and Structures. Academic Press Limited.

9. Vandiver, J. K. And Li, Li (1994). "Suppression of Cable Vibration by Means of Wave Absorbing Terminations". Unpublished.

10. Vandiver, J. K. And Li, Li. "Vibration Suppression of Tension Dominated Slender Structures by Dynamic Absorbers". Unpublished.

11. Vandiver, J. K. And Li, Li (1995). "Wave Propagation in Strings with Rigid Bodies". Journal of Vibration and Acoustics.

12. Vandiver, J. K. And Li, Li. "A Comparison Study of the Green's Function and Mode Superposition Techniques and Their Application to the Lock-in Response Prediction of Cylinders in Currents". Unpublished.

13. Griffin, O. M. (1985). "Vortex Induced Vibrations of Marine Cables and Structures". NRL Memorandum Report 5600. Naval Research Laboratory.

14. Conte, S. D. And de Boor, C. (1980). Elementary Numerical Analysis. McGraw-Hill Book Company, New York.

Appendix A: MATLAB Files

This appendix contains all MATLAB files used in this thesis. The files are divided into four directories, each corresponding to a different cable system model. The following is a list of these directories and a description of the system they model:

CENTER: A cable system 50 meters long with an in line bending type absorber at its center.

OFFCENTER: A cable system 50 meters long with an in line bending type absorber located at 7L/16.

MULTABS: A cable system 50 meters long with three in line bending type absorbers, each located at a mode 3 antinode.

HANG: A cable system 50 meters long with a hanging mass spring dashpot absorber located at its center.

Variables and Dimensions

The following variables are used throughout the MATLAB programs in this Appendix:

Variable	Corresponding Parameter	Units
a, b, c, d, e, f	cable dimensions	m
M1, M2	absorber link masses	kg
T, H	cable tension	N
K	stiffness	N/m
R	damping constant	N/(m/s)
I1, I2	absorber link moments of inertia	kg m^2
p	cable mass density	kg/m
c	cable wave propagation speed	m/s
k	cable wave number	m^{-1}
w	circular frequency	rad/s

CENTER

The following is a typical sequence for using the model CENTER to determine system natural frequencies and mode shapes and optimize system damping:

1) In MATLAB, change directory to CENTER.

2) Edit STRING1 for desired absorber properties and STRING2 for desired cable properties.

3) In STRING4 and STRING1, ensure damping constant is set to zero. Run STRING4 to plot t_{12} of $[T_{overall}]$ versus frequency. The zeroes of this plot are the undamped system natural frequencies.

4) If desired, plot undamped system mode shapes as follows:

   a) For mode of interest, enter natural frequency in MATLAB window.

   b) Run STRING6 to plot mode shape.

   c) Repeat for other modes of interest.

5) Perform absorber optimization for individual modes as follows:

   a) Edit MULRUN to iterate over desired range of damping constants.

   b) Also in MULRUN, set **zrs** to the undamped natural frequency for the mode of interest. This serves as a starting guess for the complex root solver.

6) Set damping constant to global in STRING1. Run MULRUN to plot real and imaginary natural frequency and damping ratio versus damping constant. The optimum damping constant is that which maximizes damping ratio.

7) Plot damped system mode shape as follows:

   a) Set damping constant to the optimum value in STRING1.

   b) Enter the complex natural frequency in the MATLAB window.

   c) Run STRING6 to plot mode shape.

8) Repeat steps 5 through 7 for other modes of interest.

```
%                    STRING1

% This file contains the transfer matrix for a vibration
% absorber.

% Initial data

T=5000;
a=.03;
b=.03;
c=.2;
d=.2;
e=.2;
f=.2;
M1=9.0;
M2=M1;
H=T;
I1=.12;
I2=I1;
K=10;
R=4000;
h=c+d;
l=e+f;

a1=(T*d)+(H*c)+(K*a^2)-(w^2*I1)+(i*w*R*b^2);

a2=(T*f)+(H*e)+(K*a^2)-(w^2*I2)+(i*w*R*b^2);
a3=(K*a^2)+(i*w*R*b^2);

g1=(-(M1*l+M2*f)*a1-
(M1*c*a3)+(w^2*M1*M2*f*c^2))/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g2=(-(M2*f*h^2)-
(M1*l*d^2)+(a1*l+a3*h)/w^2)/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g3=-
((M1*l*d*c)+(a1*l+a3*h)/w^2)/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g4=-((M1*l+M2*f)*a3+(M1*c*a2)+(w^2*M1*M2*e*c*f))/((a2*h+a3*l)/w^2-
(M1*d*l^2+M2*f^2*h));

g5=((a2*h+a3*l)/w^2+(M2*e*f*h))/((a2*h+a3*l)/w^2-(M1*d*l^2+M2*f^2*h));

g6=(-a2*(M1*d+M2*h)-(M2*e*a3)+(w^2*M1*M2*e^2*d))/((a2*h+a3*l)/w^2-
(M1*d*l^2+M2*f^2*h));
```

```
t(1,1)=g1+g3*((g4+g1*g6)/(1-g3*g6));
t(1,2)=g2+g3*((g5+g2*g6)/(1-g3*g6));
t(2,1)=(g4+g1*g6)/(1-g3*g6);
t(2,2)=(g5+g2*g6)/(1-g3*g6);
```

```
%                    STRING2

% This file calculates the transfer matrix for a string of length L.

%Initial data

T=5000;
L=25;
p=1.0;
c=sqrt(T/p);
k=w/c;
t(1,1)=cos(k*L);
t(1,2)=sin(k*L)/(k*T);
t(2,1)=-k*T*sin(k*L);
t(2,2)=cos(k*L);
```

```
%                    STRING4

% This file plots the overall T(1,2) for a system with 3 components.
W=0;
Ttot12=0;

R=0;
for n=1:250;
w=.1*n;
W(n)=w;
string2
T1=t;
T3=t;
string1
T2=t;
Ttot=T1*T2*T3;
Ttot12(n)=Ttot(1,2);
end
plot(W,abs(Ttot12))
grid
xlabel(’w(rad/sec)’)
ylabel(’T(1,2)’)
```

```
%                         STRING5

% This file contains the transfer matrix for a uniform bar.
%  Initial data:

% bar length
lb=.8;

M=18.0;
T=5000;
Ib=M*lb*w^2/T;

k=1/(2*(6+Ib));
t(1,1)=k*4*(3-Ib);
t(1,2)=k*12*lb/T;
t(2,1)=-k*M*w^2*(12-Ib);
t(2,2)=k*4*(3-Ib);
```

```
%                       STRING6

%       THIS FILE PLOTS THE MODE SHAPES OF A STRING
%       WITH A DAMPER AT ITS CENTER.  BEFORE RUNNING,
%       ENTER FREQUENCY IN THE MATLAB WINDOW AND
%       ENTER DAMPING IN STRING1.
y=0;
f=0;
T=5000;
p=1.0;
c=sqrt(T/p);
k=w/c;
x=0:.1:50.8;

for n=1:251
tl(1,1)=cos(k*x(n));
tl(1,2)=sin(k*x(n))/(k*T);
tl(2,1)=-k*T*sin(k*x(n));
tl(2,2)=cos(k*x(n));
y(n)=tl(1,2);
f(n)=tl(2,2);
end



string1
ts(1,1)=cos(25*k);
ts(1,2)=sin(25*k)/(k*T);
ts(2,2)=cos(25*k);
ts(2,1)=-k*T*sin(25*k);

for n=258:509
tr(1,1)=cos(k*(x(n)-25.8));
tr(1,2)=sin(k*(x(n)-25.8))/(k*T);
tr(2,1)=-k*T*sin(k*(x(n)-25.8));
tr(2,2)=cos(k*(x(n)-25.8));
tf=ts*t*tr;
y(n)=tf(1,2);
f(n)=tf(2,2);
end

plot(x(1:251),real(y(1:251)),x(258:509),real(y(258:509)))
grid
xlabel('x location')
```

```
ylabel('real(displacement mode shape)')
title('3rd MODE MODE SHAPE')
```

```
%                         STRING14

% This file computes T(1,2) for a system with 3 components.

function T12=string14(w)
string2
T1=t;
T3=t;
string1
T2=t;
Ttot=T1*T2*T3;
T12=Ttot(1,2);
```

```
%                    MULLER

function [zros]=muller(fn,nprev,np,maxit,ep1,ep2,fnreal,zros);
% zeros=muller(fn,nprev,maxit,ep1,ep2,fnreal,zeros)
% This function is called by the driver program mulrun.m
% Determines up to np zeros of the function specified by fn,
% using quadratic interpolation, i.e., Muller's Method.
% This function is a translation of the Fortran routine given
% by S. D. Conte and Carl de Boor, "Elementary Numerical
% Analysis", McGraw Hill Book Company, 1980. The conversion
% was made by Howard Wilson and Chris Roberts, Engineering
% Mechanics Department, University of Alabama.
%
% To find more than one zero, function muller uses a procedure
% known as deflation, which is implemented in function dflate.
% fn: fuction that we want to find the roots.
% zros: an array of initial estimates of all desired roots, set to zero if
%     no better estimates are available.
% ep1: relative error tolerance on the root.
% ep2: error tolerance on the function.
% maxit: maximum number of iterations per root allowed.
% np: number of roots desired.
% nprev: number of roots previously computed, normally zero.
% fnreal: a logical number; if TRUE, the program forces all approximations to
%      all the roots to be real. This makes it possible to use this routine
%      even if f(x) is defined only for real x.
%      Here we can take fnreal=1, if we only want to fine the real roots; we
%      can take fnreal=0 if we want to find both the real and complex roots.
fn='string14';
eps1=max(ep1,1.e-12); eps2=max(ep2,1.e-20);
for i=nprev+1:np; kount=0;
 while 1

%  Compute the first three estimates for zero as
%  zero, zero+0.5, zero-0.5

   zero=zros(i); h=0.00001+j*0.00001; zp=zero+h; zm=zero-h;
   [fzr,dvdf1p,kount,zros]=dflate(fn,zp,i,kount,zros);
   [fzr,fzrprv,kount,zros]=dflate(fn,zm,i,kount,zros);
   hprev=-2.0*h; dvdf1p=(fzrprv-dvdf1p)/hprev;
   [fzr,fzrdfl,kount,zros]=dflate(fn,zero,i,kount,zros);
   while 2

%    Do while kount<maxit or h is relatively big or fzr=f(zero)
```

```
%         is not small or fzrdfl=fdeflated(zero) is not small or not
%         much bigger than its previous value fzrprv.

       divdf1=(fzrdfl-fzrprv)/h; divdf2=(divdf1-dvdf1p)/(h+hprev);
       hprev=h; dvdf1p=divdf1; c=divdf1+h*divdf2;
       sqr=c*c-4.0*fzrdfl*divdf2;
       if fnreal*real(sqr)<0.0, sqr=0.0; else, sqr=sqrt(sqr); end;
       if real(c)*real(sqr)+imag(c)*imag(sqr)<0, den=c-sqr;
       else, den=c+sqr; end
       if abs(den)<0.0, den=1.0; end
       h=-2.0*fzrdfl/den; fzrprv=fzrdfl; zero=zero+h;
       if kount>maxit, break, else
         while 3
           [fzr,fzrdfl,kount,zros]=dflate(fn,zero,i,kount,zros);
           if zros(i)==zero*1.001, break; end
%            Check for convergence
           if abs(h)<eps1*abs(zero), break; end
           if max(abs(fzr),abs(fzrdfl))<eps2, break; end
%            Check for divergence
           if abs(fzrdfl)>10.0*abs(fzrprv), h=h/2; zero=zero-h;
           else, break; end
        end; % while 3
      end; % end kount <= maxit
      if zros(i)==zero*1.001, break; end
      if abs(h)<eps1*abs(zero), break; end
      if max(abs(fzr),abs(fzrdfl))<eps2, break; end
    end; % while 2
    if kount>maxit, break; end
    if abs(h)<eps1*abs(zero), break; end
    if max(abs(fzr),abs(fzrdfl))<eps2, break; end
  end; % while 1
  zros(i)=zero;
end; % for loop
```

```
%                    MULRUN


% ***********************************
x=0;
y=0;
global R
% input specs for Muller
for n=1:51
R=100*(n-1);
x(n)=R;
%  tolerance, name of function that computes T12 etc:

  zrs = 121.;            % Starting guess for the root, can be real or complex
  fn = 'string14';       % name of program that computes T12
  maxit = 500;                 % max # of iterations
     ep1   =   1e-5;         %tolerance   on   the   root   (start   with   lower
                 %     values, 1e-2or  so.)
  tolfcn = 1e-5;         % relative tolerance on T12
  fnreal = 0;            % = 0 complex search, = 1, real search
  nprev = 0;             % leave these as 0
  np = 1;                %        and 1



  ep2 = tolfcn*string14(zrs); % This sets the tolerance on T12 to be
                              %   1e-5   of   T12   estimated   at   your   starting
                           % guess

% Call Muller

 root = muller(fn,nprev,np,maxit,ep1,ep2,fnreal,zrs);
y(n)=root;
end
zeta=imag(y)./real(y);

subplot(311)
plot(x,imag(y))
grid
title('4th mode')
xlabel('R')
ylabel('imag(nat freq)')
subplot(312)
plot(x,real(y))
```

```
grid
xlabel('R')
ylabel('real(nat freq)')
subplot(313)
plot(x,zeta)
grid
xlabel('R (N/(m/s))')
ylabel('modal damping ratio')
```

```
%                    DFLATE

% This is the subroutine needed by muller.
function [fzero,fzrdfl,kount,zros]=dflate(F,zero,i,kount,zros);
% (*fzero,fzrdfl,kount,zeros*)=dflate(F,zero,i,kount,zeros)
% This function which is called by function muller is translated
% from the Fortran routine given by S. D. Conte and Carl de Boor,
% "Elementary Numerical Analysis", McGraw Hill Book Company, 1980.
% The translation was made by Howard Wilson and Chris Roberts,
% Engineering Mechanics Department, University of Alabama.

kount=kount+1; fzero=feval(F,zero); fzrdfl=fzero;
if i<2, return; end
for j=2:i
  den=zero-zros(j-1);
  if abs(den)==0.0
    zros(i)=zero*1.001; return
  else
   fzrdfl=fzrdfl/den;
  end
end
```

OFFCENTER

The following is a typical sequence for using the model OFFCENTER to determine system natural frequencies and mode shapes and optimize system damping:

1) In MATLAB, change directory to OFFCENTER.

2) Edit STRING1A for desired absorber properties and STRING2AL and STRING2AR for desired cable properties.

3) In STRING4A and STRING1A, ensure damping constant is set to zero. Run STRING4A to plot t_{12} of [$T_{overall}$] versus frequency. The zeroes of this plot are the undamped system natural frequencies.

4) If desired, plot undamped system mode shapes as follows:

   a) For mode of interest, enter natural frequency in MATLAB window.

   b) Run STRING6A to plot mode shape.

   c) Repeat for other modes of interest.

5) Perform absorber optimization for individual modes as follows:

   a) Edit MULRUN to iterate over desired range of damping constants.

   b) Also in MULRUN, set **zrs** to the undamped natural frequency for the mode of interest. This serves as a starting guess for the complex root solver.

6) Set damping constant to global in STRING1A. Run MULRUN to plot real and imaginary natural frequency and damping ratio versus damping constant. The optimum damping constant is that which maximizes damping ratio.

7) Plot damped system mode shape as follows:

   a) Set damping constant to the optimum value in STRING1A.

   b) Enter the complex natural frequency in the MATLAB window.

   c) Run STRING6A to plot mode shape.

8) Repeat steps 5 through 7 for other modes of interest.

```
%                               STRING1A

% This file contains the transfer matrix for a vibration
% absorber.

% Initial data

T=5000;
a=.03;
b=.03;
c=.2;
d=.2;
e=.2;
f=.2;
M1=9;
M2=M1;
H=5000;
I1=.12;
I2=I1;
K=10;
R=100000;
h=c+d;
l=e+f;

a1=(T*d)+(H*c)+(K*a^2)-(w^2*I1)+(i*w*R*b^2);

a2=(T*f)+(H*e)+(K*a^2)-(w^2*I2)+(i*w*R*b^2);
a3=(K*a^2)+(i*w*R*b^2);

g1=(-(M1*l+M2*f)*a1-
(M1*c*a3)+(w^2*M1*M2*f*c^2))/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g2=(-(M2*f*h^2)-
(M1*l*d^2)+(a1*l+a3*h)/w^2)/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g3=-
((M1*l*d*c)+(a1*l+a3*h)/w^2)/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g4=-((M1*l+M2*f)*a3+(M1*c*a2)+(w^2*M1*M2*e*c*f))/((a2*h+a3*l)/w^2-
(M1*d*l^2+M2*f^2*h));

g5=((a2*h+a3*l)/w^2+(M2*e*f*h))/((a2*h+a3*l)/w^2-(M1*d*l^2+M2*f^2*h));

g6=(-a2*(M1*d+M2*h)-(M2*e*a3)+(w^2*M1*M2*e^2*d))/((a2*h+a3*l)/w^2-
(M1*d*l^2+M2*f^2*h));
```

```
t(1,1)=g1+g3*((g4+g1*g6)/(1-g3*g6));
t(1,2)=g2+g3*((g5+g2*g6)/(1-g3*g6));
t(2,1)=(g4+g1*g6)/(1-g3*g6);
t(2,2)=(g5+g2*g6)/(1-g3*g6);
```

```
%      STRING2AL

%  This file calculates the transfer matrix for a string of
     length L,
%   located on the left side of the absorber

%Initial data
T=5000;
L=(7/16)*50;
p=1.0;
c=sqrt(T/p);
k=w/c;
t(1,1)=cos(k*L);
t(1,2)=sin(k*L)/(k*T);
t(2,1)=-k*T*sin(k*L);
t(2,2)=cos(k*L);
```

```
%                    STRING2AR

% This file calculates the transfer matrix for a string of length L,
%  located on the right side of the absorber.

%Initial data

T=5000;
L=(9/16)*50;
p=1.0;
c=sqrt(T/p);
k=w/c;
t(1,1)=cos(k*L);
t(1,2)=sin(k*L)/(k*T);
t(2,1)=-k*T*sin(k*L);
t(2,2)=cos(k*L);
```

```
%                    STRING4A

% This file plots the overall T(1,2) for a system with 3 components.
W=0;
Ttot12=0;

R=0;
for n=1:200;
w=.1*n;
W(n)=w;
string2al
T1=t;
string2ar
T3=t;
string1a
T2=t;
Ttot=T1*T2*T3;
Ttot12(n)=Ttot(1,2);
end
plot(W,abs(Ttot12))
grid
xlabel('w(rad/sec)')
ylabel('T(1,2)')
```

```
%                       STRING6A

%       THIS FILE PLOTS THE MODE SHAPES OF A STRING
%       WITH AN OFF CENTER DAMPER.  ENTER FREQ.
%       FROM THE MATLAB WINDOW AND DAMPING
%       IN STRING1A
x=0;
y=0;
T=5000;
p=1.0;
c=sqrt(T/p);
k=w/c;
x=0:.1:50.8;

for n=1:220
tl(1,1)=cos(k*x(n));
tl(1,2)=sin(k*x(n))/(k*T);
tl(2,1)=-k*T*sin(k*x(n));
tl(2,2)=cos(k*x(n));
y(n)=tl(1,2);
f(n)=tl(2,2);
end



string1a
ts(1,1)=cos(21.9*k);
ts(1,2)=sin(21.9*k)/(k*T);
ts(2,2)=cos(21.9*k);
ts(2,1)=-k*T*sin(21.9*k);

for n=227:509
tr(1,1)=cos(k*(x(n)-22.7));
tr(1,2)=sin(k*(x(n)-22.7))/(k*T);
tr(2,1)=-k*T*sin(k*(x(n)-22.7));
tr(2,2)=cos(k*(x(n)-22.7));
tf=ts*t*tr;
y(n)=tf(1,2);
f(n)=tf(2,2);
end

plot(x(1:220),real(y(1:220)),x(227:509),real(y(227:509)))
grid
xlabel('x location')
```

```
ylabel('real(displacement mode shape)')
title('4TH MODE MODE SHAPE, DAMPER AT 3L/8')
```

```
%                    STRING14A

% This file computes T(1,2) for a cable with an off center absorber.

function T12=string14a(w)
string2al
T1=t;
string2ar
T3=t;
string1a
T2=t;
Ttot=T1*T2*T3;
T12=Ttot(1,2);
```

```
%                    MULLER

function [zros]=muller(fn,nprev,np,maxit,ep1,ep2,fnreal,zros);
% zeros=muller(fn,nprev,maxit,ep1,ep2,fnreal,zeros)
% This function is called by the driver program mulrun.m
% Determines up to np zeros of the function specified by fn,
% using quadratic interpolation, i.e., Muller's Method.
% This function is a translation of the Fortran routine given
% by S. D. Conte and Carl de Boor, "Elementary Numerical
% Analysis", McGraw Hill Book Company, 1980. The conversion
% was made by Howard Wilson and Chris Roberts, Engineering
% Mechanics Department, University of Alabama.
%
% To find more than one zero, function muller uses a procedure
% known as deflation, which is implemented in function dflate.
% fn: fuction that we want to find the roots.
% zros: an array of initial estimates of all desired roots, set to zero if
%     no better estimates are available.
% ep1: relative error tolerance on the root.
% ep2: error tolerance on the function.
% maxit: maximum number of iterations per root allowed.
% np: number of roots desired.
% nprev: number of roots previously computed, normally zero.
% fnreal: a logical number; if TRUE, the program forces all approximations to
%       all the roots to be real. This makes it possible to use this routine
%       even if f(x) is defined only for real x.
%       Here we can take fnreal=1, if we only want to fine the real roots; we
%       can take fnreal=0 if we want to find both the real and complex roots.
fn='string14a';
eps1=max(ep1,1.e-12); eps2=max(ep2,1.e-20);
for i=nprev+1:np; kount=0;
 while 1

%  Compute the first three estimates for zero as
%  zero, zero+0.5, zero-0.5

  zero=zros(i); h=0.00001+j*0.00001; zp=zero+h; zm=zero-h;
  [fzr,dvdf1p,kount,zros]=dflate(fn,zp,i,kount,zros);
  [fzr,fzrprv,kount,zros]=dflate(fn,zm,i,kount,zros);
  hprev=-2.0*h; dvdf1p=(fzrprv-dvdf1p)/hprev;
  [fzr,fzrdfl,kount,zros]=dflate(fn,zero,i,kount,zros);
  while 2

%   Do while kount<maxit or h is relatively big or fzr=f(zero)
```

```
%    is not small or fzrdfl=fdeflated(zero) is not small or not
%    much bigger than its previous value fzrprv.

      divdf1=(fzrdfl-fzrprv)/h; divdf2=(divdf1-dvdf1p)/(h+hprev);
      hprev=h; dvdf1p=divdf1; c=divdf1+h*divdf2;
      sqr=c*c-4.0*fzrdfl*divdf2;
      if fnreal*real(sqr)<0.0, sqr=0.0; else, sqr=sqrt(sqr); end;
      if real(c)*real(sqr)+imag(c)*imag(sqr)<0, den=c-sqr;
      else, den=c+sqr; end
      if abs(den)<0.0, den=1.0; end
      h=-2.0*fzrdfl/den; fzrprv=fzrdfl; zero=zero+h;
      if kount>maxit, break, else
       while 3
        [fzr,fzrdfl,kount,zros]=dflate(fn,zero,i,kount,zros);
        if zros(i)==zero*1.001, break; end
%        Check for convergence
        if abs(h)<eps1*abs(zero), break; end
        if max(abs(fzr),abs(fzrdfl))<eps2, break; end
%        Check for divergence
        if abs(fzrdfl)>10.0*abs(fzrprv), h=h/2; zero=zero-h;
        else, break; end
       end; % while 3
      end; % end kount <= maxit
      if zros(i)==zero*1.001, break; end
      if abs(h)<eps1*abs(zero), break; end
      if max(abs(fzr),abs(fzrdfl))<eps2, break; end
    end; % while 2
    if kount>maxit, break; end
    if abs(h)<eps1*abs(zero), break; end
    if max(abs(fzr),abs(fzrdfl))<eps2, break; end
  end; % while 1
  zros(i)=zero;
end; % for loop
```

```
%                         MULRUN


% ************************************
x=0;
y=0;
global R
%  input specs for Muller
for n=1:21
R=10000*(n-1);
x(n)=R;
%   tolerance, name of function that computes T12 etc:

  zrs = 10.88;          % Starting guess for the root, can be real or complex
  fn = 'string14a';     % name of program that computes T12
  maxit = 500;                  % max # of iterations
     ep1    =    1e-5;          %tolerance    on    the    root    (start    with    lower
                    %      values, 1e-2or  so.)
  tolfcn = 1e-5;        % relative tolerance on T12
  fnreal = 0;           % = 0 complex search, = 1, real search
  nprev = 0;            % leave these as 0
  np = 1;               %          and 1



  ep2 = tolfcn*string14a(zrs);       % This sets the tolerance on T12 to be
                              %    1e-5    of    T12    estimated    at    your    starting
                          % guess

% Call Muller

 root = muller(fn,nprev,np,maxit,ep1,ep2,fnreal,zrs);
y(n)=root;
end
zeta=imag(y)./real(y);

subplot(311)
plot(x,imag(y))
grid
title('4th mode')
xlabel('R')
ylabel('imag(nat freq)')
subplot(312)
plot(x,real(y))
```

```
grid
xlabel('R')
ylabel('real(nat freq)')
subplot(313)
plot(x,zeta)
grid
xlabel('R')
ylabel('modal damping ratio')
```

```
%                     DFLATE

% This is the subroutine needed by muller.
function [fzero,fzrdfl,kount,zros]=dflate(F,zero,i,kount,zros);
% (*fzero,fzrdfl,kount,zeros*)=dflate(F,zero,i,kount,zeros)
% This function which is called by function muller is translated
% from the Fortran routine given by S. D. Conte and Carl de Boor,
% "Elementary Numerical Analysis", McGraw Hill Book Company, 1980.
% The translation was made by Howard Wilson and Chris Roberts,
% Engineering Mechanics Department, University of Alabama.

kount=kount+1; fzero=feval(F,zero); fzrdfl=fzero;
if i<2, return; end
for j=2:i
 den=zero-zros(j-1);
 if abs(den)==0.0
   zros(i)=zero*1.001; return
 else
  fzrdfl=fzrdfl/den;
 end
end
```

MULTABS

The following is a typical sequence for using the model MULTABS to determine system natural frequencies and mode shapes and optimize system damping:

1) In MATLAB, change directory to MULTABS.

2) Edit STRING1A for desired absorber properties and STRING2AI and STRING2AO for desired cable properties.

3) In STRING4A and STRING1A, ensure damping constant is set to zero. Run STRING4A to plot t_{12} of [$T_{overall}$] versus frequency. The zeroes of this plot are the undamped system natural frequencies.

4) If desired, plot undamped system mode shapes as follows:

    a) For mode of interest, enter natural frequency in MATLAB window.

    b) Run STRING6A to plot mode shape.

    c) Repeat for other modes of interest.

5) Perform absorber optimization for individual modes as follows:

    a) Edit MULRUN to iterate over desired range of damping constants.

    b) Also in MULRUN, set **zrs** to the undamped natural frequency for the mode of interest. This serves as a starting guess for the complex root solver.

6) Set damping constant to global in STRING1A. Run MULRUN to plot real and imaginary natural frequency and damping ratio versus damping constant. The optimum damping constant is that which maximizes damping ratio.

7) Plot damped system mode shape as follows:

    a) Set damping constant to the optimum value in STRING1A.

    b) Enter the complex natural frequency in the MATLAB window.

    c) Run STRING6A to plot mode shape.

8) Repeat steps 5 through 7 for other modes of interest.

```
%                    STRING1A

% This file contains the transfer matrix for a vibration
% absorber.

% Initial data

T=5000;
a=.03;
b=.03;
c=.2;
d=.2;
e=.2;
f=.2;
M1=9;
M2=M1;
H=5000;
I1=.12;
I2=I1;
K=10;
global R;
h=c+d;
l=e+f;

a1=(T*d)+(H*c)+(K*a^2)-(w^2*I1)+(i*w*R*b^2);

a2=(T*f)+(H*e)+(K*a^2)-(w^2*I2)+(i*w*R*b^2);
a3=(K*a^2)+(i*w*R*b^2);

g1=(-(M1*l+M2*f)*a1-
(M1*c*a3)+(w^2*M1*M2*f*c^2))/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g2=(-(M2*f*h^2)-
(M1*l*d^2)+(a1*l+a3*h)/w^2)/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g3=-
((M1*l*d*c)+(a1*l+a3*h)/w^2)/((M2*e*a1)+(M1*d+M2*h)*a3+(w^2*M1*M2*c*e*d));

g4=-((M1*l+M2*f)*a3+(M1*c*a2)+(w^2*M1*M2*e*c*f))/((a2*h+a3*l)/w^2-
(M1*d*l^2+M2*f^2*h));

g5=((a2*h+a3*l)/w^2+(M2*e*f*h))/((a2*h+a3*l)/w^2-(M1*d*l^2+M2*f^2*h));

g6=(-a2*(M1*d+M2*h)-(M2*e*a3)+(w^2*M1*M2*e^2*d))/((a2*h+a3*l)/w^2-
(M1*d*l^2+M2*f^2*h));
```

```
t(1,1)=g1+g3*((g4+g1*g6)/(1-g3*g6));
t(1,2)=g2+g3*((g5+g2*g6)/(1-g3*g6));
t(2,1)=(g4+g1*g6)/(1-g3*g6);
t(2,2)=(g5+g2*g6)/(1-g3*g6);
```

```
%                    STRING2AI

% This file calculates the transfer matrix for a string of length L/3,
%  located on the left side of the abosorber

%Initial data
T=5000;
L=(1/3)*50;
p=1.0;
c=sqrt(T/p);
k=w/c;
t(1,1)=cos(k*L);
t(1,2)=sin(k*L)/(k*T);
t(2,1)=-k*T*sin(k*L);
t(2,2)=cos(k*L);
```

```
%                    STRING2AO

% This file calculates the transfer matrix for a string of length L/6,
%  located on the left side of the abosorber

%Initial data
T=5000;
L=(1/6)*50;
p=1.0;
c=sqrt(T/p);
k=w/c;
t(1,1)=cos(k*L);
t(1,2)=sin(k*L)/(k*T);
t(2,1)=-k*T*sin(k*L);
t(2,2)=cos(k*L);
```

```
%                     STRING4A

% This file plots the overall T(1,2) for a system with absorbers located
% at the mode 3 antinodes.  Before running, set R to the desired value
% in STRING1A.

W=0;
Ttot12=0;

R=0;
for n=1:100;
w=100+.1*n;
W(n)=w;
string2ao
T1=t;
string2ai
T3=t;
string1a
T2=t;
Ttot=T1*T2*T3*T2*T3*T2*T1;
Ttot12(n)=Ttot(1,2);
end
plot(W,abs(Ttot12))
grid
xlabel('w(rad/sec)')
ylabel('T(1,2)')
```

```
%                         STRING6A

%       THIS FILE PLOTS THE MODE SHAPES OF A STRING
%       WITH ABSORBERS AT THE MODE 3 ANTINODES.
%        ENTER FREQUENCY
%       FROM THE MATLAB WINDOW AND DAMPING
%       IN STRING1A
x=0;
y=0;
T=5000;
p=1.0;
c=sqrt(T/p);
k=w/c;
x=0:.1:52.4;

for n=1:84
tl(1,1)=cos(k*x(n));
tl(1,2)=sin(k*x(n))/(k*T);
tl(2,1)=-k*T*sin(k*x(n));
tl(2,2)=cos(k*x(n));
y(n)=tl(1,2);
f(n)=tl(2,2);
end

string1a


tso(1,1)=cos(8.33*k);
tso(1,2)=sin(8.33*k)/(k*T);
tso(2,2)=cos(8.33*k);
tso(2,1)=-k*T*sin(8.33*k);

tsi(1,1)=cos(16.67*k);
tsi(1,2)=sin(16.67*k)/(k*T);
tsi(2,2)=cos(16.67*k);
tsi(2,1)=-k*T*sin(16.67*k);

for n=92:259
tlc(1,1)=cos(k*(x(n)-9.1));
tlc(1,2)=sin(k*(x(n)-9.1))/(k*T);
tlc(2,1)=-k*T*sin(k*(x(n)-9.1));
tlc(2,2)=cos(k*(x(n)-9.1));
tf=tso*t*tlc;
y(n)=tf(1,2);
```

```
f(n)=tf(2,2);
end


for n=267:434
trc(1,1)=cos(k*(x(n)-26.6));
trc(1,2)=sin(k*(x(n)-26.6))/(k*T);
trc(2,1)=-k*T*sin(k*(x(n)-26.6));
trc(2,2)=cos(k*(x(n)-26.6));
tf=tso*t*tsi*t*trc;
y(n)=tf(1,2);
f(n)=tf(2,2);
end

for n=442:525
tr(1,1)=cos(k*(x(n)-44.1));
tr(1,2)=sin(k*(x(n)-44.1))/(k*T);
tr(2,1)=-k*T*sin(k*(x(n)-44.1));
tr(2,2)=cos(k*(x(n)-44.1));
tf=tso*t*tsi*t*tsi*t*tr;
y(n)=tf(1,2);
f(n)=tf(2,2);
end
plot(x(1:84),real(y(1:84)),x(92:259),real(y(92:259)),x(267:434),real(y(267:434)),x(442:525),real(y(442:5
grid
xlabel(’x location’)
ylabel(’real(displacement mode shape)’)
title(’3rd MODE MODE SHAPE, DAMPERS AT ANTINODES’)
```

```
%                    STRING14A

% This file computes T(1,2) for a cable with absorbers located at the
%  mode 3 antinodes.

function T12=string14a(w)
string2ao
T1=t;
string2ai
T3=t;
string1a
T2=t;

Ttot=T1*T2*T3*T2*T3*T2*T1;
T12=Ttot(1,2);
```

```
%                    MULLER

function [zros]=muller(fn,nprev,np,maxit,ep1,ep2,fnreal,zros);
% zeros=muller(fn,nprev,maxit,ep1,ep2,fnreal,zeros)
% This function is called by the driver program mulrun.m
% Determines up to np zeros of the function specified by fn,
% using quadratic interpolation, i.e., Muller's Method.
% This function is a translation of the Fortran routine given
% by S. D. Conte and Carl de Boor, "Elementary Numerical
% Analysis", McGraw Hill Book Company, 1980. The conversion
% was made by Howard Wilson and Chris Roberts, Engineering
% Mechanics Department, University of Alabama.
%
% To find more than one zero, function muller uses a procedure
% known as deflation, which is implemented in function dflate.
% fn: fuction that we want to find the roots.
% zros: an array of initial estimates of all desired roots, set to zero if
%      no better estimates are available.
% ep1: relative error tolerance on the root.
% ep2: error tolerance on the function.
% maxit: maximum number of iterations per root allowed.
% np: number of roots desired.
% nprev: number of roots previously computed, normally zero.
% fnreal: a logical number; if TRUE, the program forces all approximations to
%       all the roots to be real. This makes it possible to use this routine
%       even if f(x) is defined only for real x.
%       Here we can take fnreal=1, if we only want to fine the real roots; we
%       can take fnreal=0 if we want to find both the real and complex roots.
fn='string14a';
eps1=max(ep1,1.e-12); eps2=max(ep2,1.e-20);
for i=nprev+1:np; kount=0;
 while 1

%  Compute the first three estimates for zero as
%  zero, zero+0.5, zero-0.5

  zero=zros(i); h=0.00001+j*0.00001; zp=zero+h; zm=zero-h;
  [fzr,dvdf1p,kount,zros]=dflate(fn,zp,i,kount,zros);
  [fzr,fzrprv,kount,zros]=dflate(fn,zm,i,kount,zros);
  hprev=-2.0*h; dvdf1p=(fzrprv-dvdf1p)/hprev;
  [fzr,fzrdfl,kount,zros]=dflate(fn,zero,i,kount,zros);
  while 2

%   Do while kount<maxit or h is relatively big or fzr=f(zero)
```

```
 %   is not small or fzrdfl=fdeflated(zero) is not small or not
 %   much bigger than its previous value fzrprv.

   divdf1=(fzrdfl-fzrprv)/h; divdf2=(divdf1-dvdf1p)/(h+hprev);
   hprev=h; dvdf1p=divdf1; c=divdf1+h*divdf2;
   sqr=c*c-4.0*fzrdfl*divdf2;
   if fnreal*real(sqr)<0.0, sqr=0.0; else, sqr=sqrt(sqr); end;
   if real(c)*real(sqr)+imag(c)*imag(sqr)<0, den=c-sqr;
   else, den=c+sqr; end
   if abs(den)<0.0, den=1.0; end
   h=-2.0*fzrdfl/den; fzrprv=fzrdfl; zero=zero+h;
   if kount>maxit, break, else
    while 3
     [fzr,fzrdfl,kount,zros]=dflate(fn,zero,i,kount,zros);
     if zros(i)==zero*1.001, break; end
%      Check for convergence
     if abs(h)<eps1*abs(zero), break; end
     if max(abs(fzr),abs(fzrdfl))<eps2, break; end
%      Check for divergence
     if abs(fzrdfl)>10.0*abs(fzrprv), h=h/2; zero=zero-h;
     else, break; end
    end; % while 3
   end; % end kount <= maxit
   if zros(i)==zero*1.001, break; end
   if abs(h)<eps1*abs(zero), break; end
   if max(abs(fzr),abs(fzrdfl))<eps2, break; end
  end; % while 2
  if kount>maxit, break; end
  if abs(h)<eps1*abs(zero), break; end
  if max(abs(fzr),abs(fzrdfl))<eps2, break; end
 end; % while 1
 zros(i)=zero;
end; % for loop
```

```
%                         MULRUN


% ************************************
x=0;
y=0;
global R
% input specs for Muller
for n=1:21
R=500*(n-1);
x(n)=R;
%  tolerance, name of function that computes T12 etc:

  zrs = 104;            % Starting guess for the root, can be real or complex
  fn = 'string14a';     % name of program that computes T12
  maxit = 500;                  % max # of iterations
     ep1    =    1e-5;          %tolerance    on    the    root    (start    with    lower
                     %       values, 1e-2or  so.)
  tolfcn = 1e-5;        % relative tolerance on T12
  fnreal = 0;           % = 0 complex search, = 1, real search
  nprev = 0;            % leave these as 0
  np = 1;               %           and 1



  ep2 = tolfcn*string14a(zrs);         % This sets the tolerance on T12 to be
                                 %    1e-5    of    T12    estimated    at    your    starting
                              % guess

% Call Muller

 root = muller(fn,nprev,np,maxit,ep1,ep2,fnreal,zrs);
y(n)=root;
end
zeta=imag(y)./real(y);

subplot(311)
plot(x,imag(y))
grid
title(' mode, dampers at mode 3 antinodes')
xlabel('R')
ylabel('imag(nat freq)')
subplot(312)
plot(x,real(y))
```

```
grid
xlabel('R')
ylabel('real(nat freq)')
subplot(313)
plot(x,zeta)
grid
xlabel('R')
ylabel('modal damping ratio')
```

```
%                    DFLATE

% This is the subroutine needed by muller.
function [fzero,fzrdfl,kount,zros]=dflate(F,zero,i,kount,zros);
% (*fzero,fzrdfl,kount,zeros*)=dflate(F,zero,i,kount,zeros)
% This function which is called by function muller is translated
% from the Fortran routine given by S. D. Conte and Carl de Boor,
% "Elementary Numerical Analysis", McGraw Hill Book Company, 1980.
% The translation was made by Howard Wilson and Chris Roberts,
% Engineering Mechanics Department, University of Alabama.

kount=kount+1; fzero=feval(F,zero); fzrdfl=fzero;
if i<2, return; end
for j=2:i
  den=zero-zros(j-1);
  if abs(den)==0.0
     zros(i)=zero*1.001; return
  else
    fzrdfl=fzrdfl/den;
  end
end
```

HANG

The following is a typical sequence for using the model HANG to determine system natural frequencies and mode shapes and optimize system damping:

1) In MATLAB, change directory to HANG.

2) Edit STRINGl for desired absorber properties and STRING2 for desired cable properties.

3) In STRING4 and STRING1, ensure damping constant is set to zero. Run STRING4 to plot t_{12} of $[T_{overall}]$ versus frequency. The zeroes of this plot are the undamped system natural frequencies.

4) If desired, plot undamped system mode shapes as follows:

   a) For mode of interest, enter natural frequency in MATLAB window.

   b) Run STRING6 to plot mode shape.

   c) Repeat for other modes of interest.

5) Perform absorber optimization for individual modes as follows:

   a) Edit MULRUN to iterate over desired range of damping constants.

   b) Also in MULRUN, set **zrs** to the undamped natural frequency for the mode of interest. This serves as a starting guess for the complex root solver.

6) Set damping constant to global in STRING1. Run MULRUN to plot real and imaginary natural frequency and damping ratio versus damping constant. The optimum damping constant is that which maximizes damping ratio.

7) Plot damped system mode shape as follows:

   a) Set damping constant to the optimum value in STRING1.

   b) Enter the complex natural frequency in the MATLAB window.

   c) Run STRING6 to plot mode shape.

8) Repeat steps 5 through 7 for other modes of interest.

```
%                          STRING1

% This file contains the transfer matrix for a hanging mass spring
% dashpot vibration absorber.

% Initial data

M=5;
K=68.45;
R=15;


t(1,1)=1;
t(1,2)=0;
t(2,1)=-(w^2*M*(i*w*R+K))/(K-w^2*M+i*w*R);
t(2,2)=1;
```

```
%                        STRING2

% This file calculates the transfer matrix for a string of length L.

%Initial data

T=5000;
L=25;
p=1.0;
c=sqrt(T/p);
k=w/c;
t(1,1)=cos(k*L);
t(1,2)=sin(k*L)/(k*T);
t(2,1)=-k*T*sin(k*L);
t(2,2)=cos(k*L);
```

```
%                                STRING4

% This file plots the overall T(1,2) for a system with 3 components.
W=0;
Ttot12=0;

R=0;
for n=1:200;
w=0.1*n;
W(n)=w;
string2
T1=t;
T3=t;
string1
T2=t;
Ttot=T1*T2*T3;
Ttot12(n)=Ttot(1,2);
end
plot(W,abs(Ttot12))
grid
xlabel('w(rad/sec)')
ylabel('T(1,2)')
```

```
%                    STRING6

%      THIS FILE PLOTS THE MODE SHAPES OF A STRING
%      WITH A MASS SPRING DASHPOT DAMPER AT ITS %        CENTER.
BEFORE RUNNING,
%      ENTER FREQUENCY IN THE MATLAB WINDOW AND
%      ENTER DAMPING IN STRING1.
y=0;
f=0;
T=5000;
p=1.0;
c=sqrt(T/p);
k=w/c;
x=0:.1:50;

for n=1:251
tl(1,1)=cos(k*x(n));
tl(1,2)=sin(k*x(n))/(k*T);
tl(2,1)=-k*T*sin(k*x(n));
tl(2,2)=cos(k*x(n));
y(n)=tl(1,2);
f(n)=tl(2,2);
end



string1
ts(1,1)=cos(25*k);
ts(1,2)=sin(25*k)/(k*T);
ts(2,2)=cos(25*k);
ts(2,1)=-k*T*sin(25*k);

for n=252:501
tr(1,1)=cos(k*(x(n)-25));
tr(1,2)=sin(k*(x(n)-25))/(k*T);
tr(2,1)=-k*T*sin(k*(x(n)-25));
tr(2,2)=cos(k*(x(n)-25));
tf=ts*t*tr;
y(n)=tf(1,2);
f(n)=tf(2,2);
end

plot(x(1:251),real(y(1:251)),x(252:501),real(y(252:501)))
grid
```

```
xlabel('x location')
ylabel('real(displacement mode shape)')
title('3rd MODE MODE SHAPE')
```

```
%                       STRING14

% This file computes T(1,2) for a system with 3 components.

function T12=string14(w)
string2
T1=t;
T3=t;
string1
T2=t;
Ttot=T1*T2*T3;
T12=Ttot(1,2);
```

```
%                    MULLER

function [zros]=muller(fn,nprev,np,maxit,ep1,ep2,fnreal,zros);
% zeros=muller(fn,nprev,maxit,ep1,ep2,fnreal,zeros)
% This function is called by the driver program mulrun.m
% Determines up to np zeros of the function specified by fn,
% using quadratic interpolation, i.e., Muller's Method.
% This function is a translation of the Fortran routine given
% by S. D. Conte and Carl de Boor, "Elementary Numerical
% Analysis", McGraw Hill Book Company, 1980. The conversion
% was made by Howard Wilson and Chris Roberts, Engineering
% Mechanics Department, University of Alabama.
%
% To find more than one zero, function muller uses a procedure
% known as deflation, which is implemented in function dflate.
% fn: fuction that we want to find the roots.
% zros: an array of initial estimates of all desired roots, set to zero if
%     no better estimates are available.
% ep1: relative error tolerance on the root.
% ep2: error tolerance on the function.
% maxit: maximum number of iterations per root allowed.
% np: number of roots desired.
% nprev: number of roots previously computed, normally zero.
% fnreal: a logical number; if TRUE, the program forces all approximations to
%      all the roots to be real. This makes it possible to use this routine
%      even if f(x) is defined only for real x.
%      Here we can take fnreal=1, if we only want to fine the real roots; we
%      can take fnreal=0 if we want to find both the real and complex roots.
fn='string14';
eps1=max(ep1,1.e-12); eps2=max(ep2,1.e-20);
for i=nprev+1:np; kount=0;
 while 1

%  Compute the first three estimates for zero as
%  zero, zero+0.5, zero-0.5

  zero=zros(i); h=0.00001+j*0.00001; zp=zero+h; zm=zero-h;
  [fzr,dvdf1p,kount,zros]=dflate(fn,zp,i,kount,zros);
  [fzr,fzrprv,kount,zros]=dflate(fn,zm,i,kount,zros);
  hprev=-2.0*h; dvdf1p=(fzrprv-dvdf1p)/hprev;
  [fzr,fzrdfl,kount,zros]=dflate(fn,zero,i,kount,zros);
  while 2

%   Do while kount<maxit or h is relatively big or fzr=f(zero)
```

```
 %    is not small or fzrdfl=fdeflated(zero) is not small or not
 %    much bigger than its previous value fzrprv.

   divdf1=(fzrdfl-fzrprv)/h; divdf2=(divdf1-dvdf1p)/(h+hprev);
   hprev=h; dvdf1p=divdf1; c=divdf1+h*divdf2;
   sqr=c*c-4.0*fzrdfl*divdf2;
   if fnreal*real(sqr)<0.0, sqr=0.0; else, sqr=sqrt(sqr); end;
   if real(c)*real(sqr)+imag(c)*imag(sqr)<0, den=c-sqr;
   else, den=c+sqr; end
   if abs(den)<0.0, den=1.0; end
   h=-2.0*fzrdfl/den; fzrprv=fzrdfl; zero=zero+h;
   if kount>maxit, break, else
    while 3
     [fzr,fzrdfl,kount,zros]=dflate(fn,zero,i,kount,zros);
     if zros(i)==zero*1.001, break; end
%     Check for convergence
     if abs(h)<eps1*abs(zero), break; end
     if max(abs(fzr),abs(fzrdfl))<eps2, break; end
%     Check for divergence
     if abs(fzrdfl)>10.0*abs(fzrprv), h=h/2; zero=zero-h;
     else, break; end
    end; % while 3
   end; % end kount <= maxit
   if zros(i)==zero*1.001, break; end
   if abs(h)<eps1*abs(zero), break; end
   if max(abs(fzr),abs(fzrdfl))<eps2, break; end
  end; % while 2
  if kount>maxit, break; end
  if abs(h)<eps1*abs(zero), break; end
  if max(abs(fzr),abs(fzrdfl))<eps2, break; end
 end; % while 1
 zros(i)=zero;
end; % for loop
```

```
%                    MULRUN


% ***********************************
x=0;
y=0;
global R
% input specs for Muller
for n=1:41
R=.5*n;
x(n)=R;
%  tolerance, name of function that computes T12 etc:

  zrs = 4.07;           % Starting guess for the root, can be real or complex
  fn = 'string14';      % name of program that computes T12
  maxit = 500;                 % max # of iterations
    ep1   =   1e-5;        %tolerance   on   the   root   (start   with   lower
                     %     values, 1e-2or  so.)
  tolfcn = 1e-5;        % relative tolerance on T12
  fnreal = 0;           % = 0 complex search, = 1, real search
  nprev = 0;            % leave these as 0
  np = 1;               %        and 1



  ep2 = tolfcn*string14(zrs); % This sets the tolerance on T12 to be
                             %   1e-5   of   T12   estimated   at   your   starting
                          % guess

% Call Muller

 root = muller(fn,nprev,np,maxit,ep1,ep2,fnreal,zrs);
y(n)=root;
end
zeta=imag(y)./real(y);

subplot(311)
plot(x,imag(y))
grid
title('4th mode')
xlabel('R')
ylabel('imag(nat freq)')
subplot(312)
plot(x,real(y))
```

```
grid
xlabel('R')
ylabel('real(nat freq)')
subplot(313)
plot(x,zeta)
grid
xlabel('R')
ylabel('modal damping ratio')
```

```
%                    DFLATE

% This is the subroutine needed by muller.
function [fzero,fzrdfl,kount,zros]=dflate(F,zero,i,kount,zros);
% (*fzero,fzrdfl,kount,zeros*)=dflate(F,zero,i,kount,zeros)
% This function which is called by function muller is translated
% from the Fortran routine given by S. D. Conte and Carl de Boor,
% "Elementary Numerical Analysis", McGraw Hill Book Company, 1980.
% The translation was made by Howard Wilson and Chris Roberts,
% Engineering Mechanics Department, University of Alabama.

kount=kount+1; fzero=feval(F,zero); fzrdfl=fzero;
if i<2, return; end
for j=2:i
  den=zero-zros(j-1);
  if abs(den)==0.0
     zros(i)=zero*1.001; return
  else
    fzrdfl=fzrdfl/den;
  end
end
```

Appendix B: Transfer Matrix Library

The following appendix is a direct reproduction of the transfer matrix library derived by Li Li [5]. It is included here so that this thesis can serve as a stand alone reference on transfer matrix analysis

Appendix A

Transfer Matrix Library

Definitions of the State Vectors and Transfer Matrices: (see Section 2.1)

A.1 Strings

1) A string with varying tension

The WKB solution to a string with varying-tension (see Fig. A-1) is given in [20] as

$$\bar{y} = \frac{1}{[\rho_c T(s)]^{\frac{1}{4}}}[C_1 e^{jW} + C_2 e^{-jW}] \tag{A.1}$$

The transfer matrix can be found from this solution as

$$F_i = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \tag{A.2}$$

where:

$t_{11} = (\frac{T_{i-1}}{T_i})^{\frac{1}{4}} cosW + \frac{1}{4\omega}\frac{T'_{i-1}}{(\rho_c^2 T_{i-1} T_i)^{\frac{1}{4}}} sinW$

$t_{12} = \frac{1}{\omega(\rho_c^2 T_{i-1} T_i)^{\frac{1}{4}}} sinW$

$t_{21} = [\frac{T'_{i-1}}{4}(\frac{T_i}{T_{i-1}})^{\frac{1}{4}} - \frac{T'_i}{4}(\frac{T_{i-1}}{T_i})^{\frac{1}{4}}] cosW - [\omega(\rho_c^2 T_{i-1} T_i)^{\frac{1}{4}} + \frac{1}{16\omega}\frac{T'_{i-1} T'_i}{(\rho_c^2 T_{i-1} T_i)^{\frac{1}{4}}}] sinW$

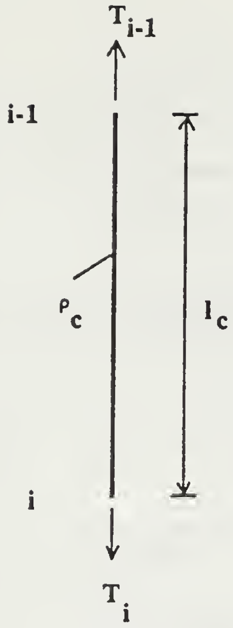


Figure A-1: A string with varying tension

$$t_{22} = \left(\frac{T_i}{T_{i-1}}\right)^{\frac{1}{4}} cosW - \frac{1}{4\omega}\frac{T_i'}{(\rho_c^2 T_{i-1} T_i)^{\frac{1}{4}}} sinW \quad .$$

where $W = \omega \int_0^{l_c} \frac{ds}{\sqrt{\frac{T(s)}{\rho_c}}}$. If the tension varies linearly, then $W = \frac{2\omega\sqrt{\rho_c}}{T'}[\sqrt{T_i} - \sqrt{T_{i-1}}]$ and $T_i' = T_{i-1}' = T'$.

2) A string with constant tension

For a string with constant tension (see Fig. A-2), with $k = \sqrt{\frac{(\rho_c\omega^2 - jR\omega)}{T}}$, the transfer matrix is

$$F_i = \begin{bmatrix} coskl_c & \frac{1}{kT} sinkl_c \\ -kT sinkl_c & coskl_c \end{bmatrix} \qquad (A.3)$$

where l_c is the distance between the two state vectors at $i-1$ and i.

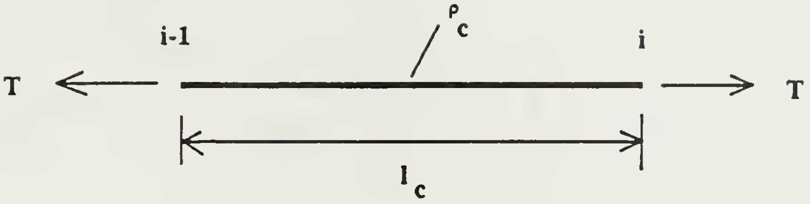


Figure A-2: A string with constant tension

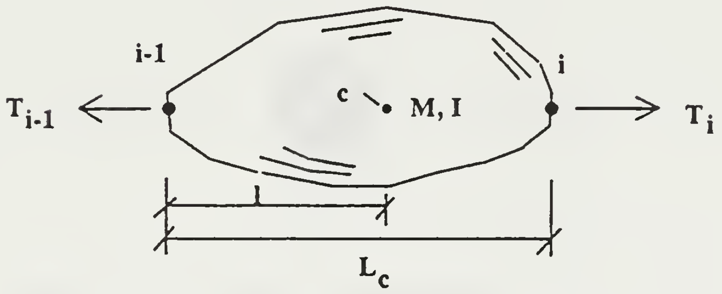


Figure A-3: A general, rigid lump

A.2 Rigid Lumps

1) A general, rigid lump

Section 2.5 gives the detailed derivation of the transfer matrix for a general, rigid lump (see Fig. A-3), the result is

$$F_i = \frac{1}{\Delta}\begin{bmatrix} T_i(L_c - l) + T_{i-1}l - M\omega^2[r^2 + (L_c - l)^2] & L_c^2 \\ -M\omega^2[T_i(L_c - l) + T_{i-1}l - M\omega^2 r^2] & T_i(L_c - l) + T_{i-1}l - M\omega^2[r^2 + l^2] \end{bmatrix} \tag{A.4}$$

where $\Delta = T_i(L_c - l) + T_{i-1}l - M\omega^2[r^2 - (L_c - l)l]$.

2) Application 1: a uniform bar

For a rigid, uniform bar (see Fig. A-4) with length l_b and linear density ρ_l, then

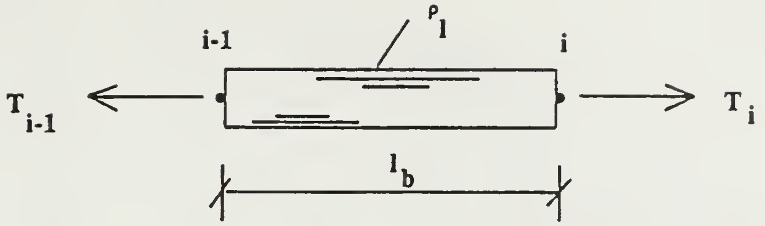


Figure A-4: A rigid, uniform bar

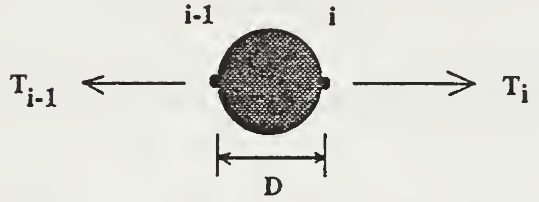


Figure A-5: A uniform, solid sphere

$L_c = l_b$, $l = \frac{l_b}{2}$, $M = \rho_l l_b$, and $r^2 = \frac{l_b^2}{12}$. The transfer matrix is

$$F_i = \frac{1}{2(6+I_b)} \begin{bmatrix} 4(3-I_b) & \frac{12l_b}{T} \\ -M\omega^2(12-I_b) & 4(3-I_b) \end{bmatrix} \tag{A.5}$$

where $I_b = \frac{Ml_b\omega^2}{T}$ and T is the average tension if there is a difference in tension at the two ends of the lump.

3) Application 2: a solid sphere

For a uniform, solid sphere (see Fig. A-5) with diameter D and volume density ρ_v, then $L_c = D$, $l = \frac{D}{2}$, $M = \frac{\pi D^3 \rho_v}{6}$, and $r^2 = \frac{D^2}{10}$. The transfer matrix is

$$F_i = \frac{1}{20+3I_s} \begin{bmatrix} 20-7I_s & \frac{20D}{T} \\ -M\omega^2(20-2I_s) & 20-7I_s \end{bmatrix} \tag{A.6}$$

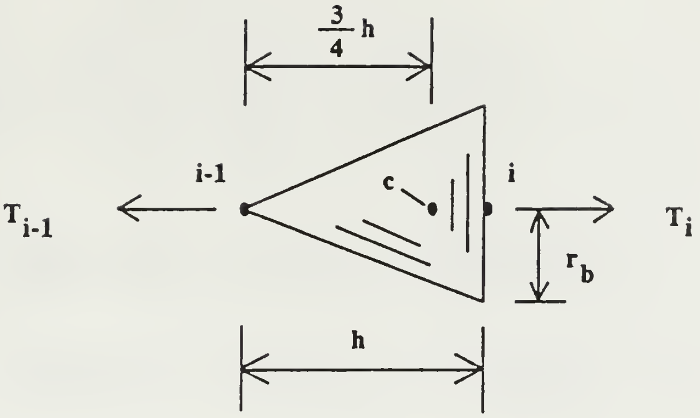


Figure A-6: A cone

where $I_s = \frac{MD\omega^2}{T}$ and T is the average tension if there is a difference in tension at two ends of the lump.

4) Application 3: a cone

For a cone (see Fig. A-6) with height h, bottom radius r_b, and volume density ρ_v, then $L_c = D$, $l = \frac{3}{4}h$, $M = \frac{\pi r_b^2 h \rho_v}{3}$, and $r^2 = \frac{3(r_b^2 + \frac{1}{4}h^2)}{20}$. The transfer matrix is

$$F_i = \frac{1}{1 - I_c(r_b^2 - h^2)} \begin{bmatrix} 1 - I_c(r_b^2 + \frac{2}{3}h^2) & \frac{4h}{T_i + 3T_{i-1}} \\ -M\omega^2[1 - I_c(r_b^2 + \frac{1}{4}h^2) & 1 - I_c(r_b^2 + 4h^2) \end{bmatrix} \tag{A.7}$$

where $I_c = \frac{3M\omega^2}{5(T_i + 3T_{i-1})h}$.

5) Application 4: a point mass

The transfer matrix for a point mass with mass M can be obtained directly from the transfer matrix for the general, rigid lump with $L_c = l = r = 0$:

$$F_i = \begin{bmatrix} 1 & 0 \\ -M\omega^2 & 1 \end{bmatrix} \tag{A.8}$$

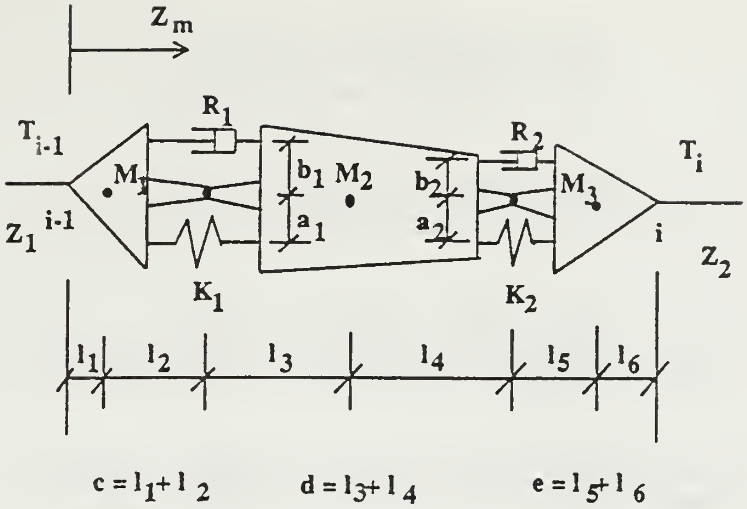


Figure A-7: A general, rigid lump with damping devices (type I)

A.3 Damping Devices

For a rigid lump with a damping device, the transfer matrix is defined the same way as that for a rigid lump without a damping device,

$$\begin{Bmatrix} \bar{y} \\ T\bar{y}' \end{Bmatrix}_i = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{Bmatrix} \bar{y} \\ T\bar{y}' \end{Bmatrix}_{i-1} \tag{A.9}$$

The expressions for the elements of the transfer matrix are as follows:

$$t_{11} = \gamma_1 + \gamma_3 \frac{\gamma_4 + \gamma_1\gamma_6}{1 - \gamma_3\gamma_6}, \quad t_{12} = \gamma_2 + \gamma_3 \frac{\gamma_5 + \gamma_2\gamma_6}{1 - \gamma_3\gamma_6}, \quad t_{21} = \frac{\gamma_4 + \gamma_1\gamma_6}{1 - \gamma_3\gamma_6}, \quad t_{22} = \frac{\gamma_5 + \gamma_2\gamma_6}{1 - \gamma_3\gamma_6} \tag{A.10}$$

For the system shown in Fig. A-7, γ's are given as follows:

$$\gamma_1 = \frac{g_8(g_1g_{12} - g_3g_{11}) - g_5(g_2g_{11} + g_1g_9)}{g_{12}(g_2g_6 - g_4g_8) + g_9(g_4g_5 + g_3g_6)} \tag{A.11}$$

$$\gamma_2 = \frac{g_5(g_9 - g_2g_{10}) - g_8(g_3g_{10} + g_{12})}{g_{12}(g_2g_6 - g_4g_8) + g_9(g_4g_5 + g_3g_6)} \tag{A.12}$$

$$\gamma_3 = \frac{g_9(g_3g_7 - g_5) + g_{12}(g_2g_7 + g_8)}{g_{12}(g_2g_6 - g_4g_8) + g_9(g_4g_5 + g_3g_6)} \tag{A.13}$$

$$\gamma_4 = \frac{g_1(g_5g_{18} - g_8g_{13}) + g_{17}(g_2g_5 + g_3g_8)}{g_{18}(g_3g_7 - g_5) + g_{13}(g_2g_7 + g_8) + g_{14}(g_2g_5 + g_3g_8)} \tag{A.14}$$

$$\gamma_5 = \frac{g_8g_{13} - g_{15}(g_2g_5 + g_3g_8) - g_5g_{18}}{g_{18}(g_3g_7 - g_5) + g_{13}(g_2g_7 + g_8) + g_{14}(g_2g_5 + g_3g_8)} \tag{A.15}$$

$$\gamma_6 = \frac{g_{13}(g_2g_6 - g_4g_8) - g_{16}(g_2g_5 + g_3g_8) + g_{18}(g_4g_5 + g_3g_6)}{g_{18}(g_3g_7 - g_5) + g_{13}(g_2g_7 + g_8) + g_{14}(g_2g_5 + g_3g_8)} \tag{A.16}$$

where g's are given by

$g_1 = \omega^2 M_1(1 - \frac{l_1}{c})$, $g_2 = \omega^2(M_1\frac{l_1}{c} + M_2\frac{l_4}{d})$, $g_3 = \omega^2(M_2\frac{l_3}{d} + M_3\frac{L_5}{e})$,

$g_4 = \omega^2 M_3(1 - \frac{l_6}{e})$, $g_5 = \frac{\alpha_3}{e} + \omega^2 M_3\frac{l_5 l_6}{e} + \frac{\beta_2}{d}$, $g_6 = \frac{\alpha_3}{e} + \omega^2 M_3 l_5(\frac{l_6}{e} - 1)$,

$g_7 = e$, $g_8 = \frac{\beta_2}{d}$, $g_9 = \frac{\alpha_1}{c} + \omega^2 M_1\frac{l_1 l_2}{c} + \frac{\beta_1}{d}$,

$g_{10} = c$, $g_{11} = \frac{\alpha_1}{c} - \omega^2 M_1 l_2(1 - \frac{l_1}{c})$, $g_{12} = \frac{\beta_1}{d}$,

$g_{13} = \frac{\alpha_2}{d} - \omega^2 M_3\frac{l_4 l_6}{e} + \frac{\beta_2}{e}$, $g_{14} = l_4$, $g_{15} = l_3$,

$g_{16} = \omega^2 M_3 l_4(1 - \frac{l_6}{e}) + \frac{\beta_2}{e}$, $g_{17} = \frac{\beta_1}{c} + \omega^2 M_1 l_3(1 - \frac{l_1}{c})$, $g_{18} = \frac{\alpha_2}{d} - \omega^2 M_1\frac{l_1 l_3}{c} + \frac{\beta_1}{c}$

$\alpha_1 = T_{i-1}l_1 + H_l l_2 + K_1 a_1^2 - \omega^2 I_1 + j\omega R_1 b_1^2$

$\alpha_2 = H_r l_4 + H_l l_3 + K_1 a_1^2 + K_2 a_2^2 + j\omega(R_1 b_1^2 + R_2 b_2^2) - \omega^2 I_2$

$\alpha_3 = T_i l_6 + H_r l_5 + K_2 a_2^2 - \omega^2 I_3 + j\omega R_2 b_2^2$

$\beta_1 = K_1 a_1^2 + j\omega R_1 b_1^2$, $\beta_2 = K_2 a_2^2 + j\omega R_2 b_2^2$

where H_l and H_r are the forces along the structural length at the left and right joint, respectively, which can be found by statics.

For the system shown in Fig. A-8, γ's are given as follows:

$$\gamma_1 = \frac{-(M_1 l + M_2 f)\alpha_1 - M_1 c\alpha_3 + \omega^2 M_1 M_2 f c^2}{M_2 e\alpha_1 + (M_1 d + M_2 h)\alpha_3 + \omega^2 M_1 M_2 ced}, \quad \gamma_2 = \frac{-M_2 f h^2 - M_1 l d^2 + (\alpha_1 l + \alpha_3 h)/\omega^2}{M_2 e\alpha_1 + (M_1 d + M_2 h)\alpha_3 + \omega^2 M_1 M_2 c} \tag{A.17}$$

$$\gamma'_{5,n} = g'_8 g_{13} + g_8 g'_{13} - g_{15}(g_2 g'_5 + g_3 g'_8) - g'_5 g_{18} - g_5 g'_{18})$$

$$\gamma'_{6,n} = g'_{13}(g_2 g_6 - g_4 g_8) + g_{13}(g_2 g'_6 - g_4 g'_8) - g'_{16}(g_2 g_5 + g_3 g_8) - g_{16}(g_2 g'_5 + g_3 g'_8)$$
$$+g'_{18}(g_4 g_5 + g_3 g_6) + g_{18}(g_4 g'_5 + g_3 g'_6)$$

$$\gamma'_{4,d} = g'_{18}(g_3 g_7 - g_5) - g_{18} g'_5 + g'_{13}(g_2 g_7 + g_8) + g_{13} g'_8 + g_{14}(g_2 g'_5 + g_3 g'_8)$$

$$\gamma'_{5,d} = \gamma'_{6,d} = \gamma'_{4,d}$$

$$g'_5 = j\omega b^2(\tfrac{1}{e} + \tfrac{1}{d}),\ g'_6 = j\omega b^2 \tfrac{1}{e},\ g'_8 = j\omega b^2 \tfrac{1}{d},$$

$$g'_9 = j\omega b^2(\tfrac{1}{c} + \tfrac{1}{d}),\ g'_{11} = j\omega b^2 \tfrac{1}{c},\ g_{12} = j\omega b^2 \tfrac{1}{d},$$

$$g'_{13} = j\omega b^2(\tfrac{2}{d} + \tfrac{1}{e}),\ g'_{16} = j\omega b^2 \tfrac{1}{e},\ g'_{17} = j\omega b^2 \tfrac{1}{c},$$

$$g'_{18} = j\omega b^2(\tfrac{2}{d} + \tfrac{1}{c})$$

For the system shown in Fig. A-8, γ's are given by

$$\gamma'_1 = \frac{j\omega b^2\{[M_1(l+c)+M_2 f]\gamma_{1,d}+[M_1 d+M_2(e+h)]\gamma_{1,n}\}}{\gamma^2_{1,d}},\ \gamma'_2 = \frac{j\omega b^2\{-\frac{l+h}{\omega^2}\gamma_{2,d}+[M_1 d+M_2(e+h)]\gamma_{2,n}\}}{\gamma^2_{2,d}}$$

$$\gamma'_3 = \frac{j\omega b^2\{\frac{l+h}{\omega^2}\gamma_{3,d}+[M_1 d+M_2(e+h)]\gamma_{3,n}\}}{\gamma^2_{3,d}},\ \gamma'_4 = \frac{j\omega b^2\{-[M_1(l+c)+M_2 f]\gamma_{4,d}-\frac{h+l}{\omega^2}\gamma_{4,n}\}}{\gamma^2_{4,d}}$$

$$\gamma'_5 = \frac{j\omega b^2[\frac{h+l}{\omega^2}(\gamma_{5,d}-\gamma_{5,n})]}{\gamma^2_{5,d}},\ \gamma'_6 = \frac{j\omega b^2\{-[M_1 d+M_2(e+h)]\gamma_{6,d}-\frac{h+l}{\omega^2}\gamma_{6,n}\}}{\gamma^2_{6,d}}$$

A.4 Supports

For the support shown in Fig. A-9, the transfer matrix is

$$F_i = \begin{bmatrix} -\frac{L_c-z}{z} & 0 \\ \frac{I_o\omega^2 - T_i(L_c-z) - T_{i-1}z}{(L_c-z)z} & -\frac{z}{L_c-z} \end{bmatrix}, \quad z \neq 0, L_c \qquad (A.26)$$

where I_o is the mass moment of inertia of the rigid lump about the pinned point.

For the support shown in Fig. A-10, the transfer matrix is

$$F_i = \begin{bmatrix} -\frac{L_c-z}{z} & 0 \\ \frac{I_o\omega^2 - T(L_c-z)}{(L_c-z)z} & -\frac{z}{L_c-z} \end{bmatrix}, \quad z \neq 0, L_c \qquad (A.27)$$

where I_o is the mass moment of inertia of the rigid lump about the pinned point.

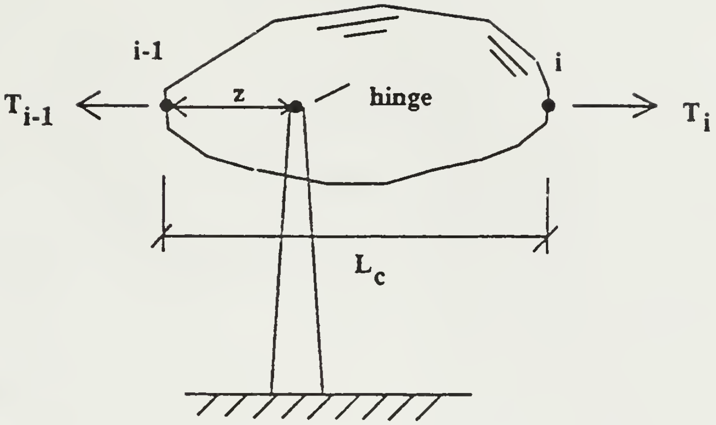


Figure A-9: An intermediate support

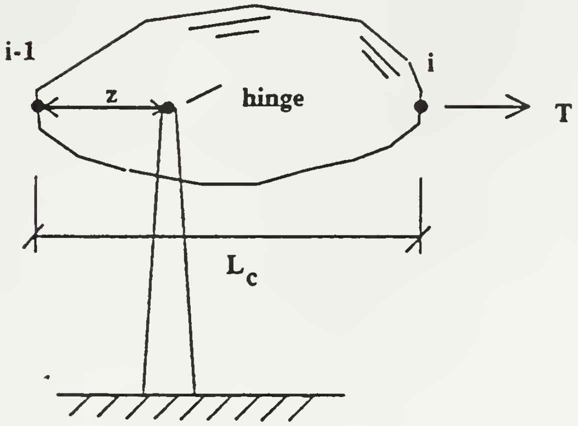
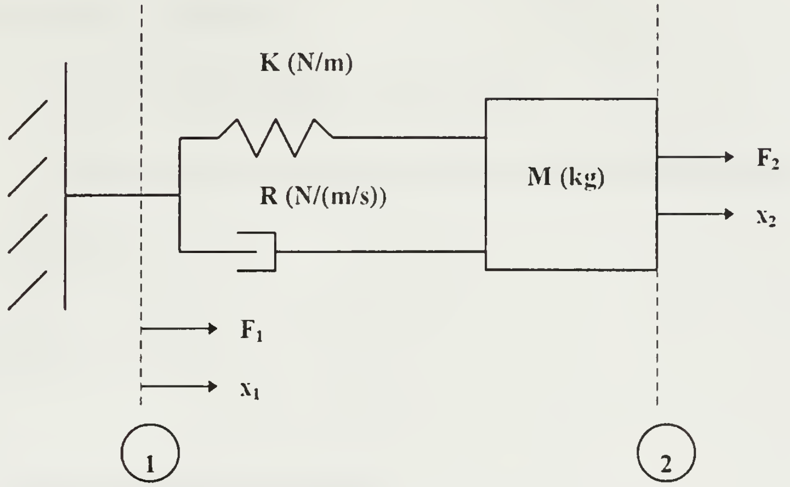


Figure A-10: An end support

Appendix C:

Analysis of a Single Degree of Freedom System Using the Transfer Matrix and Complex Natural Frequency Approach

The System:



The Transfer Matrix Relates the State Vectors at Points 1 and 2:

$$\begin{bmatrix} x_2 \\ F_2 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ F_1 \end{bmatrix} \qquad (1)$$

Derive the Transfer Matrix of the System Between Points 1 and 2:

$$-F_1 = K(x_2 - x_1) + j\omega R(x_2 - x_1) = (K + j\omega R)(x_2 - x_1)$$

solving for x_2:

$$x_2 = -F_1/(K + j\omega R) + x_1 \qquad (2)$$

$$F_1 + F_2 = M(d^2x_2/dt^2)$$

$$F_1 + F_2 = -\omega^2 M x_2$$

writing F_2 in terms of F_1 and x_1:

$$F_2 = -F_1 - \omega^2 M(-F_1/(K + j\omega R) + x_1)$$

$$F_2 = -\omega^2 M x_1 + (\omega^2 M/(K + j\omega R) - 1)F_1 \qquad (3)$$

Equations 2 and 3 result in the transfer matrix:

$$\begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} = \begin{bmatrix} 1 & -1/(K+j\omega R) \\ -\omega^2 M & -1+\omega^2 M/(K+j\omega R) \end{bmatrix}$$

Applying Boundary Conditions:

$x_1 = 0$ due to system being attached to wall

$F_2 = 0$ because we want to study the free response of the system

so, equation 1 becomes:

$$\begin{bmatrix} x_2 \\ 0 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} 0 \\ F_1 \end{bmatrix}$$

yields the characteristic equation:

$T_{22} = 0$

$-1+\omega^2 M/(K+j\omega R) = 0$

$$-\omega^2 M + j\omega R + K = 0 \qquad (4)$$

The Complex Frequency Root:

applying quadratic formula with $4KM < R^2$:

$$\omega = \frac{\sqrt{4KM - R^2}}{2M} + j\frac{R}{2M} = \omega_{real} + j\omega_{imaginary} \qquad (5)$$

From Section 3.3.1 of thesis, a damped harmonic response can be written:

$x(t) = A\cos(\omega_{real}\, t)\, e^{-\omega imag\, t}$

$$x(t) = A\cos\left(\frac{\sqrt{4KM - R^2}}{2M}t\right)\exp\left(-\frac{R}{2M}t\right) \qquad (6)$$

Comparison With Traditional Method:

In the traditional method, the characteristic equation is the same as equation 4:

$$-\omega^2 M + j\omega R + K = 0$$

but the complex root is not solved for. Instead the following are defined:

$$\omega_0 = \sqrt{\frac{K}{M}}\text{, the undamped natural frequency (7)}$$

$$\zeta = \frac{R}{2\omega_0 M}\text{, the damping ratio} \qquad (8)$$

$$\omega_d = \omega_0 \sqrt{1-\zeta^2}\text{ , the damped natural frequency} \qquad (9)$$

The displacement solution is then written as:

$$x(t) = A \cos(\omega_d t)\exp(-\zeta\omega_0 t) \qquad (10)$$

When equations 7, 8 and 9 are used to write equation 10 in terms of K, R and M. The result is:

$$x(t) = A \cos\left(\frac{\sqrt{4KM - R^2}}{2M}t\right)\exp\left(-\frac{R}{2M}t\right) \qquad (11)$$

which is the same as equation 6, the result obtained using the complex frequency approach.